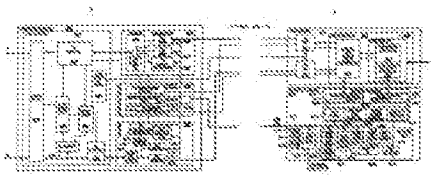
METHOD AND APPARATUS FOR TRANSFERRING DATA FROM FIRST PROTOCOL TO A SECOND PROTOCOL

Publication number: JP2002512766 (T)
Publication date: 2002-04-23
Inventor(s):
Applicant(s):
Classification:
- international: ***G06F13/12; G06F13/36; H04L12/56; H04L29/06;*** **G06F13/12; G06F13/36; H04L12/56; H04L29/06;** (IPC1-7): G06F13/12; G06F13/36; H04L29/06
- European: H04L12/56Q1; H04L29/06
Application number: JP19990550553T 19990329
Priority number(s): US19980054649 19980403; WO1999US06772 19990329

Also published as:
WO9952253 (A1)
US6185620 (B1)
EP0986886 (A1)

Abstract not available for JP 2002512766 (T)
Abstract of corresponding document: **WO 9952253 (A1)**

A method and apparatus for transferring data from a host to a node through a fabric connecting the host to the node. A chip architecture is provided in which a protocol engine provides for on ship processing in transferring data such that frequent interrupts from various components within the chip may be processed without intervention from the host processor. Additionally, context managers are provided to transmit and receive data. The protocol engine creates a list of transmit activities, which is traversed by the context managers, which in turn execute the listed activity in a fashion independent from the protocol engine. In receiving data, the context managers provide a mechanism to process frames of data originating from various sources without requiring intervention from the protocol engine. When receiving data, the context managers are able to process frames from different sources, which arrive out of order. Additionally, the context managers also determine when all frames within a sequence have been received. A link control unit is provided in which loop management is provided when the host is connected to a loop. Management of the loop includes implementing mechanisms to initiate acquisition of the loop and initiate a release of the loop in response to conditions in which data is received and transmitted by the host and by other nodes on the loop.



(19)日本国特許庁（ＪＰ）　(12) **公表特許公報（Ａ）**　(11)特許出願公表番号
特表2002－512766
（P2002－512766A）
(43)公表日　平成14年４月23日(2002.4.23)

(51)Int.Cl.[7]		識別記号	FI		テーマコード(参考)
Ｈ０４Ｌ	29/06		Ｇ０６Ｆ	13/12	３３０Ａ
Ｇ０６Ｆ	13/12	３３０		13/36	３２０Ａ
	13/36	３２０	Ｈ０４Ｌ	13/00	３０５Ｂ

審査請求　未請求　予備審査請求　有　（全 73 頁）

(21)出願番号　特願平11－550553
(86)（22)出願日　平成11年３月29日(1999.3.29)
(85)翻訳文提出日　平成11年12月３日(1999.12.3)
(86)国際出願番号　ＰＣＴ／ＵＳ９９／０６７７２
(87)国際公開番号　ＷＯ９９／５２２５３
(87)国際公開日　平成11年10月14日(1999.10.14)
(31)優先権主張番号　０９／０５４，８４９
(32)優先日　平成10年４月３日(1998.4.3)
(33)優先権主張国　米国（ＵＳ）
(81)指定国　ＥＰ(ＡＴ，ＢＥ，ＣＨ，ＣＹ，ＤＥ，ＤＫ，ＥＳ，ＦＩ，ＦＲ，ＧＢ，ＧＲ，ＩＥ，ＩＴ，ＬＵ，ＭＣ，ＮＬ，ＰＴ，ＳＥ)，ＪＰ

(71)出願人　エルエスアイ　ロジック　コーポレーション
アメリカ合衆国カリフォルニア州95035，ミルピタス，マッカーシー・ブールバード　1551
(72)発明者　ウェーバー，デーヴィッド・エム
アメリカ合衆国コロラド州80132，モニュメント，レインジ・ビュー・ロード　3680
(72)発明者　ホーグランド，ティモシー・イー
アメリカ合衆国コロラド州80918，コロラド・スプリングズ，バーゲトリー・ドライブ　3035
(74)代理人　弁理士　社本　一夫　（外５名）

最終頁に続く

(54)【発明の名称】　第１のプロトコルから第２のプロトコルへのデータ転送方法及び装置

(57)【要約】
ホストをノードに接続するファブリックを介してデータをホストからノードまで転送する方法及び装置である。チップ・アーキテクチャが提供されており、そこでは、データの転送の際にプロトコル・エンジンがオンチップ処理を提供し、チップ内の様々な要素からの頻繁な割り込みが、ホスト・プロセッサからの介入なしに処理される。更に、データの送受信のためのコンテキスト・マネジャが提供される。プロトコル・エンジンは、送信アクテビティのリストを作成し、このリストはコンテキスト・マネジャによって検討され、コンテキスト・マネジャは、リスト化されたアクテビティをプロトコル・エンジンとは独立な態様で実行する。データを受信する際には、コンテキスト・マネジャは、様々なソースからのデータのフレームを、プロトコル・エンジンからの介入を要求することなく処理するメカニズムを提供する。データを受信するときには、コンテキスト・マネジャは、異なるソースから順不同で到着するフレームを処理することができる。更に、コンテキスト・マネジャは、あるシーケンス内のすべてのフレームがいつ受信されたかを判

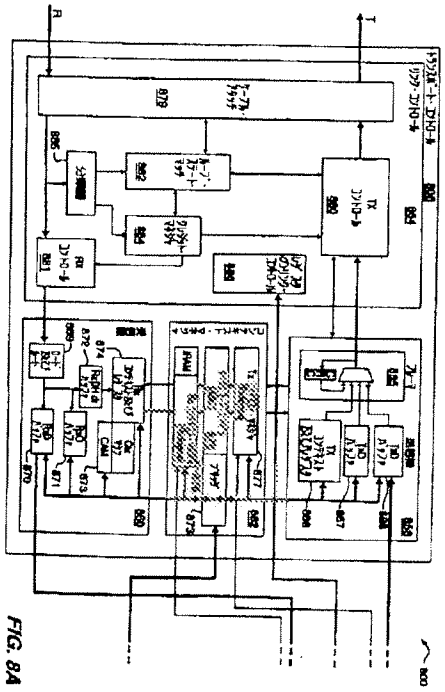


FIG. 8A

【特許請求の範囲】

１．チップであって、

入力ポートと、

出力ポートと、

第１の処理ユニットであって、

このチップから離れて位置するホストからのリクエストを受信し、データを宛先に送信する第１の受信手段と、

前記データを受信する第２の受信手段と、

前記宛先と前記データとを、前記宛先への送信に先だって、このチップに接続されたメモリに記憶する記憶手段と、

を備えている第１の処理ユニットと、

宛先に送信するデータの存在を検出する検出手段を含む第２の処理ユニットと、

前記データと前記宛先とを用いて前記宛先への搬送のためのフォーマットに前記データをフォーマットするフォーマット手段と、

を備えていることを特徴とするチップ。

２．請求項１記載のチップにおいて、前記第１の処理ユニットは、宛先に送られるデータのためのアクティビティのリストを作成し、前記検出手段は、前記リストを検査し前記宛先に送信するためのデータの存在を検出することを特徴とするチップ。

３．請求項１記載のチップにおいて、予測されるデータのリストが前記第１の処理ユニットによって発生され、前記第２の処理ユニットは、

データを遠隔ソースから受信する受信手段と、

前記データを記憶する識別手段と、

前記第１の処理ユニットに、いつすべてのデータが前記遠隔ソースから受信されたかを指示する指示手段と、

を含むことを特徴とするチップ。

４．請求項１記載のチップにおいて、前記第２の処理ユニットは埋め込み型プロセッサであることを特徴とするチップ。

５．請求項１記載のチップにおいて、前記第２の処理ユニットはステート・マシンであることを特徴とするチップ。

【発明の詳細な説明】

第１のプロトコルから第２のプロトコルへのデータ転送方法及び装置

関連出願への相互参照

この出願は、"Method and Apparatus for Managing Access to a Loop in a Data Processing System"と題し、出願番号が09/054,850であって、弁護士の事件（ドケット）番号が98-090である、同じ被譲渡人に譲渡されている同時出願中の出願と関連する。この関連出願は、本出願において援用することとする。

１．技術分野

本発明は、広くは、改善されたデータ処理システムに関し、詳しくは、１つのデータ・プロトコルから別のデータ・プロトコルにデータを転送する改善された方法及び装置に関する。更に詳しくは、本発明は、ファイバ・チャネルなどのような、チャネル・プロトコルからシリアル・プロトコルにデータを転送する改善された方法及び装置を提供する。

２．関連技術の説明

ＡＮＳＩによって採用されたファイバ・チャネル・スタンダード（ＦＣＳ）は、ワークステーション、大容量記憶装置、プリンタ、ディスプレイなどに対する低コストで高速の相互接続のためのスタンダードを提供する。ファイバ・チャネル（ＦＣ）は、分散型のシステム・アーキテクチャ、イメージ集約的なローカル・ネットワーク及びクラスタにとって理想的である。ファイバ・チャネルは、媒体依存的であり、マルチ・ベンダ相互運用性（インターオペラビリティ）を提供する。

現在のファイバ・チャネルでのデータ転送速度は、それぞれの方向で１００メガバイト／秒を超えている。また、ファイバ・チャネルのデータ転送速度は、５０メガバイト／秒や２５メガバイト／秒程度のより低速にスケーリングすることができる。この技術によると、切り換えられる媒体と共有される媒体との両方に対するチャネル接続とネットワーク接続との両方をサポートするインターフェースが得られる。ファイバ・チャネルは、デバイスの相互接続を単純化し、ハードウェアのコストを減少させる。その理由は、それぞれのデバイスが、チャネル・

インターフェースとネットワーク・インターフェースとの両方に対して、ただ１つのファイバ・チャネルだけしか要求しないからである。ネットワーク、ポート間及び周辺機器のインターフェースは、任意のフォーマットのデータ転送との同じハードウェア接続を介してのアクセスが可能である。

ターゲットとソースとの間でのデータ転送に関し、入出力（Ｉ／Ｏ）プロセッサ技術の性能が急激に向上したために、より高速で、より多くの接続性を提供し、より長い距離にわたる接続を可能にするＩ／Ｏソリューションを求めて、高性能サーバ、ワークステーション、クラスタ型計算及び関連するストレージ市場に対する要求が非常に大きくなった。大容量ストレージと全二重ファイバ・チャネル・リンク上のそれ以外のプロトコルとをサポートするように設計されている高性能でインテリジェントなＩ／Ｏプロセッサであるファイバ・チャネルＩ／Ｏプロセッサは、Ｉ／Ｏ動作をサポートするのに要求されるホストＣＰＵ及びＰＣＩ帯域幅を減少させる態様でデータを移動させることが求められる。初期化、コマンド、エラー回復など、データの移動ではないアクテビティのためにＰＣＩバスなどのシステム・バス上で費やされる時間の長さを最小化することが求められる。従って、２つの異なるデータ・プロトコルの間でデータを転送する方法及び装置の改良が求められている。

３．発明の概要

本発明は、ホストから第１のノードまでホストを第１のノードに接続するバスを介し、更に、ファブリックを介して第１のノードに接続された第２のノードまでデータを転送する方法及び装置を提供する。第１のノードは、チップ・アーキテクチャを含んでおり、そこでは、データの転送の際にプロトコル・エンジンがオンチップ処理を提供し、チップ内の様々な要素からの頻繁な割り込みがホスト・プロセッサからの介入なしに処理されうるようになっている。更に、データ

の送受信のためのコンテキスト・マネジャが提供される。プロトコル・エンジンは、送信アクテビティのリストを作成し、このリストはコンテキスト・マネジャによって検討され、コンテキスト・マネジャは、リスト化されたアクテビティをプロトコル・エンジンとは独立に実行する。データを受信する際には、コンテキ

スト・マネジャは、様々なソースからのデータのフレームを、プロトコル・エンジンからの介入を要求することなく処理するメカニズムを提供する。データを受信するときには、コンテキスト・マネジャは、異なるソースから順不同で到着するフレームを処理することができる。更に、コンテキスト・マネジャは、あるシーケンス内のすべてのフレームがいつ受信されたかを判断する。

更に、本発明は、ホストがループに接続される時にループ管理が提供されるリンク制御（コントロール）ユニットを提供する。ループの管理には、ホストとループ上の他のノードとによってデータが受信及び送信される状態に応答して、ループの取得を開始しループの解放を開始するメカニズムの実現が含まれる。

4．図面の簡単な説明

本発明の特徴付けると考えられる新規な特徴は、請求の範囲に記載されている。しかし、本発明自体は、その使用、更なる目的及び効果の好適な態様と共に、実施例に関する以下の詳細な説明を次の添付の図面と共に参照することによって、最もよく理解されるはずである。

図１は、本発明の好適実施例によるファイバ・チャネルにおける５つの層を図解している。

図２Ａ及び２Ｂは、リンク制御フレームとデータ・フレームとである。

図３は、エクスチェンジ（exchange）の図と、それがいかにしてその最小の要素に分解されるかとを示している。

図４は、本発明の好適実施例によって処理されうるＳＣＳＩエクスチェンジを図解している。

図５は、本発明が実現されうるデータ処理システムを示している。

図６は、本発明の好適実施例によるデータ処理システムのブロック図である。

図７は、本発明の好適実施例によるメッセージ要求処理を図解する図である。

図８は、本発明の好適実施例が実現されうるチップのブロック図である。

図９は、本発明のシステム内でのデータ転送を図解する機能ブロック図である。

図１０は、本発明の好適実施例によるFree_List循環キュー（待ち行列）とPos

t_List循環キューとを図解している。

図１１は、本発明の好適実施例による送信コンテキスト・マネジャにおいて実現されるプロセスを図解する流れ図である。

図１２は、本発明の好適実施例による受信制御ブロックのためのフォーマットである。

図１３は、本発明の好適実施例によってコンテキスト・スイッチングを実行するのに用いられる流れ図である。

図１４は、本発明の好適実施例によるＤＭＡ開始プロセスの流れ図である。

図１５は、本発明の好適実施例によるＤＭＡ更新プロセスの流れ図である。

図１６は、本発明の好適実施例によるフレーム完成処理プロセスである。

図１７は、本発明の好適実施例によるループ管理制御のためのステート・マシンである。

図１８は、本発明の好適実施例によって解放状態（open state）のループを管理するプロセスの流れ図である。

図１９は、本発明の好適実施例によってアイドル状態でループの取得を制御するのに用いられる規則を組み入れているプロセスの流れ図である。

図２０は、本発明の好適実施例によるループ状態待機に用いられる規則を組み入れているプロセスの流れ図である。

図２１は、本発明の好適実施例による判断ウィンドウ状態における変化を扱う際に用いられる規則を組み入れているプロセスの流れ図である。

図２２は、本発明の好適実施例による判断ウィンドウ状態における変化を扱うための規則を組み入れているプロセスの流れ図である。

５．詳細な説明

ファイバ・チャネルは、インターネット・プロトコル（ＩＰ）などのネットワーキング・プロトコルやＳＣＳＩなどのチャネル・プロトコルをサポートする高性能リンクである。ファイバ・チャネルの構造は、５つの層によって定義される。図１は、ファイバ・チャネルにおける５つの層を図解している。最下位の層であるＦＣ－０は、媒体インターフェース層である。この層は、２つのデバイス間

の物理的インターフェースを定義する。この層には、ドライバと、受信機と、銅から光へのトランスデューサと、コネクタと、銅又は光ケーブル上を１３３、２６６、５３１，１０６２メガバイト／秒の速度で送信又は受信するのに必要な任意の他の低レベル関連回路とが含まれる。

１つ上の次の層は、ＦＣ－１層である。この層は、８ｂ／１０ｂの符号化／復号化と、データ統合に必要な送信プロトコルと、送信クロック及び受信クロック回復とを定義している。この層は、通常、ＦＣ－０層とＦＣ－２層とを実現しているハードウェアの間に分割される。特に、ＦＣ－０トランシーバは、クロック回復回路を含み、他方で、８ｂ／１０ｂの符号化／復号化は、ＦＣ－２の層でなされる。次の層は、ＦＣ－２層である。この層は、フレーミング・プロトコルと転送されているデータに対するフロー制御動作とを定義する。送信又は受信されているデータの意味は、ＦＣ－２に対して透過的（transparent）である。しかし、任意の与えられたフレームの組のコンテキストは、ＦＣ－２において維持される。フレーミング・プロトコルは、必要なフレームを作成し、データは、それぞれのフレームのペイロード内にパケット化される。次の層は、ＦＣ－３層である。ＦＣ－３は、複数のN_portにわたる共通のサービスを提供する。N_portは、「ノード」ポートとも称されるが、リンクのノード端部におけるファイバ・チャネルで定義されるハードウェア・エンティティ（entity）である。これらのサービスには、ストリッピング(Striping)、ハント・グループ(Hunt Groups)及びマルチキャスティング（Multicasting）が含まれる。これらのサービスすべてにより、１つのポートが複数のN_portと一度に通信することが可能になる。ＦＣにおいて定義される最上位の層は、ＦＣ－４層である。ＦＣ－４層は、既存のスタンダードのシームレスな統合を提供する。この層は、上位の層のプロトコル（ＵＬＰ）から下位の層へのマップ（mapping）を特定する。これらのＵＬＰのいくつかには、ＳＣＳＩとインターネット・プロトコル（ＩＰ）とが含まれる。これらのＵＬＰのそれぞれは、それ自身のＡＮＳＩドキュメントにおいて定義さ

れる。

ファイバ・チャネルにおいて用いられるフレームには２つのタイプがある。す

なわち、リンク制御フレームと、データ・フレームとである。リンク制御フレームは、ペイロード（payload）を含まず、データ・フレームへの応答である。データ・フレームは、ペイロード・フィールド内のデータを含むフレームである。図２Ａ及び２Ｂを参照すると、リンク制御フレーム２００とデータ・フレーム２０２とが図解されている。それぞれのフレームは、フレームの開始（start-of-frame=SOF）フィールド２０４を含み、フレームの終了(end-of-frame=EOF）フィールド２０６の順序付き集合（ordered set）で終わる。ＳＯＦとＥＯＦとを含むすべての順序付き集合は、４バイトで構成される。それぞれのフレームは、宛先（destination）及びソースＩＤ、サービスのクラス、フレームのタイプ（すなわち、ＳＣＳＩ又はＩＰ）などの事項を定義する２４バイトのヘッダ・フィールド２０８を少なくとも含む。フレーム内の最大のフィールドは、データ・フレーム２０２内にあるペイロード・フィールド２１０である。フレームは、リンク制御フレームであるならばペイロード・フィールド２１０は存在せず、データ・フレームであるならば２１１２バイトまでのペイロード・フィールドを含む。最後に、両方のタイプのフレーム共に、送信エラーの検出に用いられる周期上長検査（ＣＲＣ）フィールド２１２を含む。

ファイバ・チャネルにおいて用いられるそれ以外の構成は、シーケンスとエクスチェンジとである。次に図３を参照すると、エクスチェンジの図とそれがどのようにして最小の要素に分解されるかとが示されている。エクスチェンジ３００には、シーケンス３０２などの１つ又は複数のシーケンスが含まれる。それぞれのシーケンスは、フレーム３０４などの１つ又は複数のフレームから構成される。エクスチェンジは、典型的なＳＣＳＩＩ／Ｏを考えることによって最もよく図解される。ＳＣＳＩＩ／Ｏでは、複数のフェーズが存在し、これらがＩ／Ｏを構成する。これらのフェーズには、コマンド、データ、メッセージ及びステータスのフェーズが含まれる。

ＳＣＳＩ（ＦＣＰ）ＵＬＰのためのファイバ・チャネル・プロトコルを用いると、これらのフェーズは、残りの下位のＦＣ層にマップされうる。図４は、本発明の好適実施例によって扱われうるＳＣＳＩエクスチェンジを図解している。Ｓ

ＣＳＩエクスチェンジ４００は、コマンド・シーケンスであるＣＭＤＳＥＱ４０２と、データ・リクエスト・シーケンスであるＤＡＴＡＲＥＱＳＥＱ４０４と、データ・シーケンスであるＤＡＴＡＳＥＱ４０６と、応答シーケンスであるＲＳＰＳＥＱ４０８とを含む。

図５には、本発明が実現され得るデータ処理システムが示されている。データ処理システム５００は、ファブリック５０６を介してターゲットに接続されたイニシエータ５０２を含む。示されている例におけるファブリック５０６は、ポイント間(point-to-point)、切換型（switched）及び仲裁型（arbitrated）ループを含む様々な幾何学的形態（トポロジ）をもちうるファイバ・チャネル・ファブリックである。データ処理システム５００では、エクスチェンジのフローは、ターゲット５０４へのフレームを１つ含むコマンド・シーケンスであるＣＭＤＳＥＱ４０２をイニシエータ５０２が送出することと共に開始する。このフレーム内のペイロードは、コマンド記述子ブロック（ＣＤＢ）を含む。ターゲット５０４は、フレームを１つ含むデータ配送リクエスト・シーケンスＤＡＴＡＲＥＱＳＥＱ４０４に応答する。このフレームのペイロードは、転送準備完了応答（transfer ready response）を含む。イニシエータ５０２は、いったんこの応答を受け取ると、１つ又は複数のフレームを含むデータ・シーケンスＤＡＴＡＳＥＱ４０６の送出を開始する(ＤＡＴＡＯＵＴフェーズ)。ターゲットは、最後のフレームを受け取ると、フレームを１つ含む応答シーケンスＲＳＰＳＥＱ４０８を送出する。このシーケンスによって、ＳＣＳＩエクスチェンジは終了する。

本発明は、異なるデータ・プロトコルの間でデータを転送するシステム、アーキテクチャ及び方法を提供する。示されている例は、ＳＣＳＩプロトコルとファイバ・チャネル・プロトコルとの間でのデータ転送のためのものである。本発明は、ホストを高速ファイバ・チャネル・インターフェースに接続するのに用いられ、また、切換型ファブリック、ポイント間、そして最も重要な仲裁型のループを含むすべてのファイバ・チャネル・トポロジにおいて用いられうる。

次に、図面を、特に、図６を参照すると、本発明の好適実施例によるデータ処理システムのブロック図が示されている。データ処理システム６００は、ホスト

６０２を含み、このホストは、データ処理システム６００の１つ又は複数のＣＰＵを形成する１つ又は複数のプロセッサを含みうる。データ処理システム６００は、I_2Oスペシャル・インタレスト・グループから入手可能なインテリジェント入出力（I_2O）アーキテクチャの仕様（１９９７年３月）と共に設計されたデータ処理システムである。これは、本出願において援用する。しかし、本発明は、他のシステム・アーキテクチャを用いても実現することができる。

ホスト６０２の中のプロセッサは、例えば、３００Ｍｈｚで動作するペンティアムIIプロセッサである。このプロセッサは、米国カリフォルニア州サンタクララ所在のインテル社から入手可能である。示されている例では、一次バス６０４と二次バス６０６とはＰＣＩバスであるが、本発明は、他のタイプのバスを用いても実現が可能である。

更に図６を参照すると、データ処理システム６００は、一次入出力プラットフォーム（ＩＯＰ）６０８を含む。これは、一次バス６０４を介して、ホスト６０２に接続されている。更に、ＩＯＰ６０８は、二次バス６０６に接続され、ＰＣＩ・ＰＣＩ間のバス・ブリッジとして機能する。データ処理システム６００はまた、アダプタ６１２及び６１４を含む。二次ＩＯＰ６１０及び６１６は、I_2Oの下にあるインテリジェント・アダプタであり、二次ＩＯＰ６１０及び６１６は、入出力プロセッサを含む。アダプタ６１２及び６１４は、非インテリジェントなアダプタであり、入出力プロセッサを含まない。

本発明のシステムは、ホストからチップへリクエスト・メッセージを転送する、そして、逆にチップからホストにリプライ・メッセージを転送するメカニズムとして、リクエスト及びリプライ・メッセージ・キューを用いる。リクエストは、ホストからチップを介してデバイスまでの経路を表し、他方で、リプライは、デバイスからチップを介してホストまでの経路を表す。

リクエスト及びリプライ・メッセージ・キューは、メッセージ・フレームの予め配分されたリストであり、共有される又はホストのメモリに存在する。チップの内部的には、それぞれのキューは、それぞれが予め配分されたメッセージ・プール内にメッセージ・フレームのアドレスを含む２つのＦＩＦＯであるフリー・リストとポスト・リストとによって特徴付けられる。フリー及びポスト・リスト

は、ホストからは可視的ではないが、メッセージ・プール内で自由な及びポストされたメッセージを管理する際にチップをサポートする。

チップが初期化されるときには、ホストは、リクエスト及びリプライ・キューがどのように管理されるべきかを選択する。デフォルトでは、リクエスト・キューは、ホスト・メモリに存在しうる。オプションとして、リクエスト・キューは、ホストとチップとの間で共有されるメモリに存在する。リプライ・キューは、常に、ホスト・メモリに存在する。リクエスト・キューとリプライ・キューとの両方へのアクセスは、ＰＣＩアドレス空間にマップされた２つのレジスタを介して提供される。

図７を参照すると、本発明の好適実施例によるメッセージ・リクエスト処理を図解している図が示されている。ホスト７００は、メッセージを構築して、リクエスト／リプライ・レジスタ７０２を読み出し次の空のメッセージ・フレームのアドレスをフリーＦＩＦＯ７０４におけるメッセージ・フレーム・プールから検索することによって、フリー・メッセージ・フレームを配分する。次に、ホスト７００は、そのリクエストをリクエスト・キュー７０６におけるメッセージ・フレームに書き込む。その後で、ホスト７００は、フレームのアドレスをリクエスト／リプライ・レジスタ７０２に書き込み、このレジスタは、このリクエストをサービスのためにポストＦＩＦＯ７１０におけるチップ７０８にポストする。ホスト７００は、次に、利用可能なフリー・メッセージが尽きるまで、このプロセスを反復して、より多くのリクエストをポストする。チップ７０８は、リクエストのアドレスをリクエスト／リプライ・レジスタ７０２から読み出し、リクエスト・キュー７０６の中のアドレスにおけるメッセージを処理し、メッセージ・アドレス（この時点では、空のメッセージ・フレーム）を再びリクエスト／リプライ・レジスタ７０２に書き込むことによって、ポストされたリクエストを読み出す。ホスト７００がリクエスト／リプライ・レジスタ７０２を読み出すときにフリー・メッセージ・フレームが存在しない場合には、チップ７０８によって供給される値は、示されている例では、ＦＦＦＦ－ＦＦＦＦｈである。

リプライ・キュー７１２は、リクエスト・キュー７０６の場合と似た態様で管理される。ただし、この場合には、チップ７０８が作成側（producer）となる

点が異なる。ホスト７００は、リプライ・キュー７１２においてリプライ・メッセージ・プールを配分し、それぞれのメッセージ・フレームのアドレスをリプライ・レジスタ７１４にポストする責任を有する。チップ７０８は、リプライを送出することを望むときには、フリーＦＩＦＯ７０４における次のフリー・メッセージ・フレームのアドレスを読み出す。チップ７０８は、リプライ・メッセージ・キュー７１２におけるフレームをメッセージで満たし、フレームのアドレスをリクエスト／リプライ・レジスタ７０２にポストし、リクエスト／リプライ・レジスタ７０２は、このアドレスをポストＦＩＦＯ７１０に書き込む。チップ７０８は、このプロセスを反復することによって、複数のリプライをポストすることがある。ホスト７００は、リクエスト／リプライ・レジスタ７０２を読み出し、ポストされたリプライ・メッセージのアドレスをポストＦＩＦＯ７１０から検索する。ホストは、このメッセージをいったん消費すると、アドレス（この時点では、フリー・メッセージ・フレーム）をリクエスト／リプライ・レジスタ７０２に書き込み、リクエスト／リプライ・レジスタ７０２は、このアドレスをフリーＦＩＦＯ７０４に書き込む。ホスト７００がリクエスト／リプライ・レジスタ７０２を読み出すときにポストされたメッセージが存在しないときには、ホスト７００は、示されている例では、ＦＦＦＦ－ＦＦＦＦｈの値を受け取る。

本発明は、リクエスト及びリプライ・キューを用いて、リクエスト及びリプライをホスト・ドライバとチップとの間で転送する。ホストがこれらのキューと相互作用する態様は、パフォーマンスに影響しうる。本発明の好適実施例によって用いられるメッセージ・キューイングには、２つのモデルが存在する。データ転送のための「プッシュ・プッシュ」モデルは、リクエスト・キューをチップによって提供されるものと定義し、リプライ・キュー・メモリをホスト・メモリ内に常駐するものとして定義する。このモデルは、データをキューの中に「プッシュ」する（押し込む）ために、リクエスト又はリプライのどちらかのイニシエータを必要とする。多くの状況で、このモデルは最適ではない。

データ転送の「プル・プッシュ」モデルでは、リクエスト及びリプライ・キューがホスト・メモリ内にあることが要求される。リクエストは、バス・マスタモードで動作しているチップの中に「プル」され(引き込まれ)、他方で、リプライ

は、ホスト・メモリの中に「プッシュ」される(押し込まれる)。このモデルによれば、チップがすべてのキューイングのためにホスト・メモリを用いることが可能となる。また、このモデルによると、チップがその動作を合理化することも可能となる。その理由は、このモデルによれば、プッシュ・プッシュ・モデルの場合のように突然に作用することを強制されるのではなく、いつコマンドを処理することを望むのかを決定することができるからである。このオプションは、複数のブリッジを介して直接にＰＣＩバスにアクセスすることにより過剰なプロセッサ・オーバーヘッドを生じるようなホスト環境に最も適している。

両方の動作モード共に、ＰＣＩバスへの同じ数のアクセスを必要とし、同じレジスタ・セットを介するキュー・アクセスを提供する。チップのためのデフォルトのオプションは、「プッシュ・プッシュ」モデルである。「プル・プッシュ」モデルは、メッセージによって呼び出すことができる。

次に、図８を参照すると、本発明の好適実施例を実現することができるチップのブロック図が図解されている。チップ８００は、プロトコル・エンジン８０２と、データ移動ユニット８０４と、移動制御ユニット８０６とを含む。

プロトコル・エンジン８０２は、この分野の当業者に既知である多数の異なるタイプのプロトコル・エンジンを用いて実現され得る。示されている例では、プロトコル・エンジン・コア８０７は、３２ビットのＲＩＳＣコア８０８に基づいている。ＲＩＳＣコア８０８は、２０－３０ＭＩＰＳの性能を有している。プロトコル・エンジン８０２は、それ自身のコントローラ８１２とリード／フェッチ／ライト・ユニット８１４とを備えた埋め込み型モジュール・バス（ＥＭＢ）８１０を含む。ＥＭＢ８１０は、バス上でのモジュール相互間の通信のための標準化されたモジュール・インターフェースを提供する。ＥＭＢ８１０は、また、示されている例では、複数のバス・マスタをサポートしている。

示されている例では、プロトコル・エンジン・コア８０７は、８ＫＢの命令／データ・バッファ８１６を含み、限界コード（critical code）及びデータ構造に対するゼロ待機状態のスタティックＲＡＭ領域を提供する。割り込みコントローラ８１８とクロック／リセット・コントローラ８２０とが、プロトコル・エンジン・コア８０７の中に見出される。ＲＩＳＣコア８０８は、ＥＭＢ８１０に

接続されたコントローラ８２２を用いている。

プロトコル・エンジン８０２の中のシステム・インターフェース８２４は、コンフィギュレーション及び高順位コマンドと、パケット化されたリクエストと、ホストとチップ８００との間のリプライ・メッセージとをサポートする。システム・インターフェース８２４は、非データ転送のためのＰＣＩバス・トラフィックを最小化するように設計されている。システム・インターフェース８２４は、また、Ｉ／Ｏリクエストとリプライ・メッセージ・パケットとをホスト・メモリとチップ８００との間で転送するのにも用いられる。ＤＭＡＦＩＦＯ８２８を備えたマスタ制御８２６とスレーブＦＩＦＯ８３２を備えたスレーブ制御８３０とは、システム・インターフェース８２４の内部に存在する。ＥＭＢアタッチ８３４は、システム・インターフェース８２４のために、ＥＭＢバス８１０への接続を提供する。ＤＭＡ及びＳＲＷ制御８３６と、スレーブ・アクセス制御８３８と、メッセージＦＩＦＯ８４０とは、システム・インターフェース８２４の内部に存在する。マスタ制御及びスレーブ制御ユニットは、ホスト・インターフェースとＤＭＡＦＩＦＯ及びスレーブＦＩＦＯそれぞれとの間のデータ転送を提供する。これらのユニットは、システム・メモリへの／システム・メモリからのデータを（バス・インターフェース・ユニットを介して）これらのＦＩＦＯの中へ及びＦＩＦＯから外へバーストする。これらのＦＩＦＯからのデータは、次に、ＥＭＢアタッチ機能（ＤＭＡＦＩＦＯ）又はスレーブ・アクセス制御機能（スレーブＦＩＦＯ）を介して、ローカル・メモリに、及び、ローカル・メモリから移動される。

ＤＭＡ及びＳＲＷユニットは、バースト転送（ＤＭＡ）又はシングル・サイクル転送（ＳＲＷ）に対するデータ転送を規制する。ｍｓｇＦＩＦＯ制御ユニットは、個々のキュー要素をローカル・メモリに、及び、ローカル・メモリからリード／ライトすることを含むメッセージング・キューを実現するのに必要なハードウェアを提供する。示されているシステム・インターフェースは、この分野の当業者に既知である多数の方法で実現することができ、プロトコル・エンジン８０２と示されているものとは別のデータ移動ユニット８０４との間のインターフェースが提供される。

プロトコル・エンジン８０２は、チップ８００の外部にあるメモリへの接続を提供する外部メモリ・コントローラ８４２を含む。メモリ・コントローラ８４２は、３２ビットのプラス・パリティＤＲＡＭ（ファースト・ページ及びＥＤＯ）と、ＥＰＲＯＭと、フラッシュ（８ビット）及びシリアルＥＥＰＲＯＭとをサポートする。示されている例では、データを転送するメモリ・データ経路８４４は、ＤＲＡＭ制御８４６とフラッシュ制御８４８とによって制御される。最後に、プロトコル・エンジン８０２は、ＥＭＢレジスタ及びタイマ・ユニット８５０に位置する自由動作（free-running）タイマを含む。このタイマは、イベント・タイム・スタンピングに用いることができる。

示されている例では、英国ケンブリッジ所在であり米国テキサス州オースチンにもオフィスを有するアドバンストＲＩＳＣマシン社（Advanced RISC Machine Ltd.）から市販されているＲＩＳＣコア８０８用のＡＲＭマイクロプロセッサ・コアを用いている。

データ移動ユニット８０４は、バス・インターフェース８５２と、送信機ＤＭＡユニット８５４と、受信機ＤＭＡユニット８５６とを含む。データ移動ユニット８０４の内部の要素は、既知のバス・インターフェース・ユニットと、送信機ＤＭＡユニットと、受信機ＤＭＡユニットとを用いて実現することができる。送信機ＤＭＡユニット８５４と受信機ＤＭＡユニット８５６とは、それぞれ、送信データ転送エンジンと受信データ転送エンジンとも称される。バス・インターフェース・ユニット８５２は、示されている例では、ＰＣＩバスを介して情報を通過させるのに用いられる。その際に、インターフェース・ユニットは、ＰＣＩマスタ・インターフェース８５９とＰＣＩスレーブ・インターフェース８６２とを含む普遍的（universal）なＰＣＩインターフェース８５８を介してマスタ及びスレーブ両方のＰＣＩバス・サイクルをサポートする。バス・インターフェース・ユニット８５２の中には、アービタ（仲裁、arbiter）及びキャッシュ・サイクル・コントローラ８６３があり、プログラム可能なアービタが、プロトコル・エンジン８０２と送信機ＤＭＡユニット８５４と受信ＤＭＡユニット８５６との間での仲裁を行うのに用いられる。データ移動ユニット８０４は、送信機ＤＭＡユニット８５４における送信機バッファに向かう、又は、受信ＤＭＡユニット８

5

６における受信バッファから出る、複数の散乱・集合（scatter-gather）データ・エントリを整列させる（align）ように設計されている。この整列（アライメント）は、３２ビットのダブル・ワード境界上でのものである。結果的に、入ってくるどれかのデータが奇数のカウントを有している場合には、フィル・バイトが追加される。データ移動ユニットは、送信機ＤＭＡユニット８５４における１つの送信散乱／集合（Ｓ／Ｇ）ＦＩＦＯと、受信ＤＭＡユニット８５６における２つの受信散乱／集合（Ｓ／Ｇ）ＦＩＦＯとを含む。３つのＦＩＦＯのすべてが、３つのＳ／Ｇエントリを含む。２つのエントリは、現在のＳ／Ｇエントリであり、残りのエントリは、次の処理されるべきＳ／Ｇエントリである。データ移動ユニット８０４は、既知のＤＭＡチャネル設計を用いて実現され得る。

移動制御ユニット８０６は、送信機８５８と、受信機８６０と、コンテキスト・マネジャ８６２と、リンク制御ユニット８６４とを含む。送信機８５８の内部には、ＴｘＤバッファ８６６とＴｘＯバッファ８６７と、送信コンテキスト及びレジスタ８６８とへの入力を備えたフレーマ８６５がある。受信機８６０は、リンク制御ユニット８６４に接続されたロード及びルート・ユニット８６９を含む。受信機８６０の中には、ＲｘＤＨバッファ８７２と、ＣｔｘマシンＣＡＭ８７３と、受信コンテキスト及びレジスタ８７４とも存在する。コンテキスト・マネジャ８６２は、コンテキスト・マネジャ８６２をＥＭＢ８１０に接続するブリッジ８７５と、送信機８５８と、受信機８６０とを含む。マイクロコード・エンジン８７６は、送信コンテキスト・マネジャ８７７と受信コンテキスト・マネジャ８７８とを制御する。コンテキスト・マネジャ８６２は、プロトコル・エンジン８０２をフリー・アップ（free up）して他の機能を実行するデータ転送機能を提供する。送信機８５８は、状態及びコンフィギュレーションのためのレジスタ（送信機コンテキスト及びレジスタ８６８）と、データ・ストレージ・バッファ（ＴｘＤバッファ８６６及びレジスタ８６８）と、フレーマ８６５とを含む領域に対する機能を提供する。フレーマ８６５は、データをバッファから取り出し、コンフィギュレーション・レジスタからの任意の要求される情報を加えて、法的

（legal）なファイバ・チャネル・フレームを発生する責任を有する。この情報は、ＳＯＦと、ヘッダと、ペイロードと、ＣＲＣと、ＥＯＦとを含む。フレーマ８６５は、次に、リンク制御ユニット８６４へのリクエストをアサートし、フレームを正しい宛先に送る。

フレーマ８６５は、フレームの全体が送られるまでリンク制御ユニット８６４のリクエストに基づいてデータを搬送する。フレーマ８６５は、また、データが２つのバッファであるＴｘＤバッファ８６６及びＴｘＯバッファ８６７から搬送する責任を有しており、それによって、バッファがロードされる。フレーマ８６５のＣＲＣサブブロックは、ファイバ・チャネルの仕様によって定義されるエラー・チェッキング・コードを計算する責任も有している。フレーマ８６５は、計算されたコードをデータ・ストリームの正しい地点に挿入する。ＴｘＤバッファ８６６は、ＰＣＩバス及びデータ移動ユニット８０４を介してホストからロードされたデータを含む。ＴｘＯバッファ８６７は、プロトコル・エンジンを介してロードされたデータを含む。このバッファは、ヘッダとＳＯＦとＥＯＦとペイロードとを備え予めフォーマットされたフレームを含む。

受信機８６０は、このノードにアドレス指定されたフレームを取り出し、そのフレームの正しさをチエックし、そのフレームを正しいメモリの宛先に配分することを助けるという責任を有する。ロード及びルート・ユニット８６９におけるロード及びルート機能は、ヘッダにおけるあるフィールドを解析して、データをどのバッファに向けて経路決定すべきかを判断する。この判断は、トラフィック・タイプに基づく（すなわち、ＳＣＳＩコマンド、ＳＣＳＩデータ、Ｉｆ）。送信側と同じように、ＲｘＯバッファ８７１は、プロトコル・エンジン８０２によって処理されるように決められたフレームを含む。やはり、フレーム全体（ヘッダ及びペイロード）が、ＲｘＯバッファ８７１に含まれる。

ＲｘＤバッファ８７０に向けられたフレームは、そのヘッダ情報が取り除かれ、別個のＲｘＤＨバッファ８７２に配置される。ＣＴｘマシンであるＣＡＭ８７３は、ヘッダ情報を用いてこのフレームがどのデータ転送に属するのかを判断する。このブロックでは、ＣＲＣもチェックされて、無効なＣＲＣを伴うフレーム

は廃棄される。ＣＴｘマシンであるＣＡＭ８７３は、最後のフレームからヘッダ情報を比較して、それがシーケンスの中の次のフレームであるかどうかを判断する。ｒｘａｑｍｅが適切なｓ／ｇエントリを備えたＤＭＡチャネルを提供するように

と要求される場合には、ＤＭＡチャネルは、データをバッファから除去する。ＲｘＤＨバッファ８７２が別個のバッファであり、それによって、先のフレームが除去されているときに後のフレームからのヘッダを解析することができる。

リンク制御ユニット８６４は、ケーブル・アタッチ８７９を含むが、これは、ファイバ・チャネルへの接続を提供し、ファイバ・チャネルからのデータを送信及び受信する。送信（ＴＸ）制御ユニット８８０と受信制御ユニット８８１とは、リンク制御ユニット８６４において見出される。ＴＸ制御ユニット８８０は、送信機８５８からのデータを受信し、そのデータをケーブル・アタッチ８７９を介してファイバ・チャネルの上に送る。ＴＸ制御ユニット８８０は、規則を適用して、フレームを送信することが望ましくかつ許容されるかを判断する。選択された時間においてフレームを転送することが望ましくなく、許容されない場合には、ＴＸ制御ユニット８８０は、フレームの送信を許容するのに必要な作用を実行する。ケーブル・アタック８７９は、８ｂ／１０ｂ符号化／復号化機能を提供し、選択された外部のシリアライザ／デシリアライザと互換な適切な態様でバイトを再度順序付ける。データは、受信制御ユニット８８１によって受信され、受信機８６０に向けて送信される。リンク制御ユニット８６４はまた、ループ・ステート・マシン８８２と、分類器（classifier）８８３と、クレジット・マネジャ８８４と、リンク制御レジスタ８８５とを含む。ループ・ステート・マシン８８２は、仲裁、送信及び受信プロトコルを含むループ関係機能を管理する。クレジット・マネジャ８８４は、フレーム・ベースのクレジット・プロトコルをモニタし管理する責任を有している。クレジット・マネジャ８８４は、別のノードがフレームをクレジット・マネジャ８８４が位置しているこのノードに送られることを可能にするにはいつクレジットが与えられるべきであるかについてトラッキングする。クレジット・マネジャ８８４は、また、フレームの送信を可能にする

のに十分なクレジットがいつ入手可能となるかについてもトラッキングする。任意の時点で、ノードは、その現在の宛先に送ることができる最大数のフレームを有する。これをクレジットと称する。ノードは、フレームを送信するときは常に、１クレジットを用いる。受信側のノードは、これらのフレームを限定されたバッファ・プールの中に受信する。フレームがバッファ・プールから除去されるとき

には、R_RDYと称されるプリミティブが発生される。送信側のノードは、R_RDYを受信するときには、そのクレジット・カウントをインクリメントする。

リンク制御レジスタ８８６は、リンク制御８６４のためのコンフィギュレーション及びステータス報告レジスタを含む。分類器８８５は、ループからの入来ワードをモニタし、多くのタイプのプリミティブの符号化を他のブロックに提供する。分類器８８５は、信号を、ループ状態マシン８８２とクレジット・マネジャ８８４とに提供する。分類器８８５におけるこの機能が提供されるのは、多くのブロックが同じプリミティブに反応からであり、復号の複写は、この状況では不要である。受信制御ユニット８８１はループ状態マシン８８２をモニタし、そのループ上を送信されているフレームはいつ受信制御ユニット８８１が位置しているノードに向けられるのかを決定する。以下でより詳細に説明するＴＸ制御ユニット８８０を除き、リンク制御ユニット８６４内の要素は、米国規格協会（ＡＮＳＩ）からのＦＣ－ＡＬ仕様を用いて、この分野の当業者に既知である要素を用いて実現することができる。

次に図９を参照すると、本発明のシステム内でのデータ転送を図解する機能ブロック図が示されている。図９は、本発明のシステム・アーキテクチャを構成する基本的な機能ブロックを示している。システム９００の中には、２つの主要なグループにグループ分けできる複数の機能ブロックが示されている。すなわち、出ていく送信グループと入ってくる受信グループとである。ブロックの間の数字の付いた矢印は、本発明のアーキテクチャによって提供される機能におけるシーケンシャルなステップを表している。この図のそれぞれの側には、メモリ要素がある。すなわち、システム・メモリ９０２と、ローカル・メモリ９０４と、シス

テム・メモリ９０６と、ローカル・メモリ９０８とであり、これらは、リクエスト・メッセージ・フレームとリプライ・メッセージ・フレームとを管理するのに用いられるフリー循環キューとポスト･フリー循環キューとを含む。リクエスト・メッセージ・フレーム構造は、図９の左側のメモリであるローカル・メモリ９０４の中にあり、リプライ・メッセージ・フレーム構造は、右側のメモリであるローカル・メモリ９０８の中にある。図の中央には、送信及び受信データ経路があり、これらは、後で、図９を参照して更に詳細に説明される。オペレーティング・

システム・モジュール（ＯＳＭ）９１０と、Ｉ／Ｏプラットフォーム（ＩＯＰ）ドライバ９１２と、メッセージ・トランスポート・マネジャ９１４と、インターフェース・マネジャ９１６と、プロトコル・フィルタ９１８とが、ブロックの２つの組として図解され、出ていく送信グループと入ってくる受信グループとの関係でのそれぞれの役割が明確に示されている。

リクエストがＯＳＭ９１０から受信されると、ＩＯＰドライバ９１２は、次のリクエスト・メッセージ・フレームのための次の空のメッセージ・フレームのアドレスを取得する(ステップＡ１)。ＩＯＰドライバ９１２は、Free_List循環キューの中のHead_Pointerに記憶されている空のメッセージ・フレーム・アドレス(EMF_ADR)を検索することによって、これを行う。すると、Head_Pointerが、インクリメントされる。図１０は、本発明の好適実施例によるFree_List循環キュー１０００とPost_List循環キュー１００２とを図解している。

ＩＯＰドライバ９１２は、検索された空のメッセージ・フレーム・アドレスにリクエスト・メッセージ・フレームを記憶する(ステップＡ２)。次に、ＩＯＰドライバ９１２は、メッセージ・フレーム・アドレスを、Post_List循環キューの中に位置するTail_Pointerの位置に記憶する。すると、Tail_Pointerは、インクリメントされる(ステップＡ３)。ＩＯＰドライバ９１２は、メッセージ・トランスポート・マネジャに、処理すべきリクエスト・メッセージ・フレームが存在することを告知する。このメカニズムは、レジスタ／割り込みベースの作用である。

次に、メッセージ・トランスポート・マネジャ９１４が、リクエスト・メッセージ・フレームのアドレスを、システム・メモリ９０２の中にあるPost_List循環キューの中のHead_Pointerから受け取る(Ａ５)。メッセージ・トランスポート・マネジャは、１つ又は複数のリクエスト・メッセージ・フレーム・アドレスを、ローカル・メモリに記憶し、リクエスト・メッセージ・フレームを処理することができる。このオプションにより、いくつかのシステムのパフォーマンスを改善することができる。次に、メッセージ・トランスポート・マネジャ９１４は、リクエスト・メッセージ・フレーム・アドレスを受け取った後で、システム･メモリ９０２の中のPost_List循環キューにおいてHead_Pointerをインクリメントさせる(ステップＡ６)。その後で、メッセージ・トランスポート・マネジャ９１４は、インターフェース・マネジャに告知し、リクエスト・メッセージ・っふれーむ・アドレスを提供する(ステップＡ７)。それに応答して、インターフェース・マネジャ９１６は、リクエスト・メッセージ・フレームのアドレス及びサイズと用いて、ＩＯＰシステムＤＭＡ９２０をプログラムする(ステップＡ８)。

ＩＯＰシステムＤＭＡ９２０は、ＰＣＩバスのために仲裁を行い、リクエスト・メッセージ・フレームをローカル・メモリの中に移動させる(ステップＡ９)。インターフェース・マネジャ９１６は、ＩＯＰシステムＤＭＡ９２０に、１つ又は複数のメッセージ・フレームを、それがFree_List循環キューの中のTail_Pointerを更新する前に、ローカル・メモリの中に移動させることができる。このオプションは、いくつかのシステムのパフォーマンスを改善することができる。次に、インターフェース・マネジャ９１６は、メッセージ・トランスポート・マネジャ９１４に、１つ又は複数のリクエスト・メッセージ・フレームをローカル・メモリ９０４の中に移動させたことを告知する(ステップ１０)。

メッセージ・トランスポート・マネジャ９１４は、新たなリクエスト・メッセージ・フレームのアドレスを、Free_List循環キューの中のTail_Pointerの位置に配置し、次に、Tail_Pointerをインクリメントする(ステップＡ１１)。これを行うことによって、リクエスト・メッセージ・フレームは、空のメッセージ・フ

レーム・リソースに変換され戻される。結果的に、インターフェース・マネジャ９１６は、プロトコル・フィルタ９１８を助け、処理のためのリクエスト・メッセージ・フレームが入手可能であることを告知する(ステップＡ１２)。プロトコル・フィルタは、ローカル・メモリ９０４からのそれぞれのリクエスト・メッセージ・フレームを検索して処理する(ステップＡ１３)。

プロトコル・フィルタ９１８は、リクエスト・メッセージ・フレームの中の情報を用いて、１つ又は複数の送信コンテキスト・ブロック（ＴＣＢ）を構築する(ステップＡ１４)。ＴＣＢは、ローカル・メモリ９２２に記憶され、送信コンテキスト・マネジャ９２４によって用いられる。いくつかのリクエスト・メッセージ・フレームは、エクスチェンジを構築するのに十分な情報を含んでいる。構築

するエクスチェンジを管理するのは、プロトコル・フィルタ９１８の仕事である。これらのエクスチェンジは、単純なログイン・エクスチェンジ又はより複雑なＳＣＳＩＩ／Ｏであり、コマンド、データ、Transfer_Rdy、応答シーケンスなどを含む。

送信コンテキスト・マネジャ９２４は、使用可能になると常にＴＣＢを送信する(ステップＡ１５)。送信コンテキスト・マネジャ９２４は、どのようなエクスチェンジ情報も知っておらず、フレーム及びシーケンス・コンテキストだけを知っている。送信コンテキスト・マネジャ９２４は、ローカル・メモリ９０４から、最上位のＴＣＢを選択し、フレーマ９２６に必要なコンテキスト情報を作成するだけでなく、必要な散乱／集合（Ｓ／Ｇ）エントリを送信Ｓ／ＧＦＩＦＯに提供する。送信コンテキスト・マネジャ９２４は、Ｓ／ＧエントリをＳ／ＧＦＩＦＯ９２８の中に配置し、コンテキスト情報をフレーマ９２６の中の配置する(ステップＡ１６)。また、ステップＡ１６において適切なときに、送信ＤＭＡ(TX_DMA)９３０が、Ｓ／ＧＦＩＦＯ９２８の最上位にあるＳ／Ｇエントリのアドレス及びサイズを用いてプログラムされ、ＰＣＩバスを求めて仲裁を行い、システム・メモリからデータを受け取り、それを送信バッファ９３２の中に記憶する。

すべてのフレームの中のすべてのデータがＴＸ＿ＤＭＡ経路から来るとは限らないことに注意することは重要である。例えば、ログイン・フレームに含まれる

１１６バイトのデータは、送信コンテキスト・マネジャ９２４を介してＤＭＡバッファ９３２の中に配置される。送信コンテキスト・マネジャ９２４は、ローカル・メモリ９０４からログイン・データを検索することによってこれを行う。ＰＣＩバス上を移動することは全く必要ない。これは、明らかなパフォーマンス上の効果である。ＴＸ＿ＤＭＡ経路を用いない別の明らかなタイプのフレームは、リンク制御フレームである。リンク制御フレームの全体は、１つのＴＣＢの中に含ませることが可能であり、送信コンテキスト・マネジャ９２４は、このＴＣＢを受信すると、それを単純に、フレーマ９２６に向けて経路決定する。フレーマ（Framer）９２６は、このデータ及びコンテキスト情報を用いて、リンク・コントローラ９３４のための１つ又は複数のフレームを作成する。

リンク・コントローラ９３４は、２つのポートの間のリンクを管理する。例えば、フレームがリンクを介して送り出される準備ができているときには、リンク・コントローラ９３４は、ループのための仲裁を行い（ＦＣ仲裁型ループ・トポロジを想定する）、仲裁を獲得すると、別の宛先であるＮＬ＿Ｐｏｒｔを開き、フレームをそこまで送る。

いったんＴＣＢとそれに関連するフレームが送信されると、送信コンテキスト・マネジャ９２４は、プロトコル・フィルタ９１８に告知する（ステップＡ１７）。プロトコル・フィルタ９１８は、ＴＣＢエントリを更新して、現在のＴＣＢの完成を反映させる（ステップＡ１８）。ＴＣＢエントリがリンクされたリストにおいてリンクされている場合には、プロトコル・フィルタ９１８は、リンクされたリストにおけるポインタを調整することによって、完成したＴＣＢエントリを除去することができる。

リンク上を送信されているすべてのデータは、結果的には、別のポートにとっては受信データとなる。図９に図解されているポート９３６に入るデータは、最初に、ギガボー・リンク・モジュール（ＧＬＭ）９３８を介して入る。このデータは、リンク・コントローラ９３４まで送られ、そこで、早期の宛先認識が生じる。入ってくるフレームがポート９３６に対するものである場合には、リンク・コントローラ９３４は、このフレームをフレーム・ディテクタ９４０に送る。フ

レームがいったんフレーム・ディテクタ９４０によって検出されると、ヘッダが取り外され、コンテキスト情報がＲＸコンテキスト・マネジャ９４２によって作成される(ステップ１９)。入ってくるフレームは、送信されたＴＣＢに対する応答か、又は、未処理のフレームでありうる。フレームが送信されたＴＣＢへの応答である場合には、コンテキストは、既に、プロトコル・フィルタによるエクスチェンジ管理の状態によって定義されている。システム・メモリに向けられた任意のデータは、受信バッファ（RX_Buffer）に入れられ、そのデータに対するＳ／ＧエントリがＳ／ＧＦＩＦＯ９２８の中に配置される(ステップＡ２０)。

フレームが未処理のフレームである場合には、コンテキストが発生されることが必要となる。最も単純なケースは、ＳＣＳＩ相互ロック・エクスチェンジの場合のように、フレームがコマンド情報を含む場合である。ＲＸコンテキスト・マネジャ９４２は、ペイロードにおけるＦＣフレーム・ヘッダとＳＣＳＩコマンド

とから、必要な情報を作成する。ステップＡ２０では、このフレームがデータも含む場合には、このデータは、RX_Bufferに配置され、Ｓ／Ｇエントリは、Ｓ／ＧＦＩＦＯ９２８の中に入れられる。データがいったんRX_Buffer９４４の中にあり、Ｓ／ＧエントリがＳ／ＧＦＩＦＯ９２８の中にあるときには、受信ＤＭＡ（ＲＸ＿ＤＭＡ）９４６は、Ｓ／Ｇのアドレス及びサイズを用いてプログラムされる。ＲＸ＿ＤＭＡ９４６は、次に、ＰＣＩバスを求めて仲裁を行い、データを、このバスを介して、システム・メモリに転送する。

コンテキスト情報は、ＲＸコンテキスト・マネジャ９４２からプロトコル・フィルタ９１８に送られ(ステップＡ２１)、そこで、コンテキストに応じて複数の事柄が生じ得る。プロトコル・フィルタ９１８又はＲＸコンテキスト・マネジャ９４２がリンク応答フレームを送る必要があるときには、常に、これは、ＴＣＢエントリをＴＣＢリンクされたリストの最上位に追加することによってなされる(ステップＡ２２)。この例は、プロトコル・フィルタ９１８が１つ又は複数の受信フレームのためにＡＣＫフレームを発生する必要があるときである。

プロトコル・フィルタ９１８は、リクエストされたメッセージ・フレームが完成されたことを告知されると、ＯＳＭのためのリプライ・メッセージを作成し、

プロトコル・フィルタ９１８は、リプライ・メッセージをローカル・メモリ９０４の中に配置する(ステップＡ２２ｂ)。

コンテキスト及びプロトコル情報を作成するのに十分な情報を含む入来の任意の未処理（unsolicited）のフレームによって、プロトコル・フィルタ９１８は、受信コンテキスト・ブロック（ＲＣＢ）が構築される(ステップＡ２２ｃ)。この例は、本発明のシステムがＳＣＳＩターゲット・モードにあるかどうかである。コマンド記述子ブロックを含むフレームが入ってくるときには、プロトコル・フィルタ９１８は、ＲＣＢリンクされたリストを発生し、ターゲットの視点からエクスチェンジ状態を管理することが必要となる。

プロトコル・フィルタ９１８は、いったん１つ又は複数のリプライ・メッセージ・フレームを作成すると、インターフェース・マネジャ９１６に、ＯＳＭ９１０に送られる必要があるリプライ・メッセージ・フレームが存在していることを告知する(ステップＡ２３)。インターフェース・マネジャ９１６は、次に、メッセージ・トランスポート・マネジャ９１４にリクエストを送り、リプライ・メッセージ・フレームのためのアドレスを取得する(ステップＡ２４)。メッセージ・トランスポート・マネジャ９１４は、リプライ・メッセージ・フレームのための次の空のメッセージ・フレームのアドレスを取得する(ステップＡ２５)。ステップＡ２５では、メッセージ・トランスポート・マネジャ９１４は、システム・メモリにあるFree_List循環キュー内のHead_Pointerに記憶されている空のメッセージ・フレーム・アドレスを検索することによってこれを行う。次に、Head_Pointerがインクリメントされる。

メッセージ・トランスポート・マネジャ９１４は、この空のメッセージ・フレーム・アドレスと共に、インターフェース・マネジャ９１６を提供する(ステップＡ２６)。インターフェース・マネジャ９１６は、リプライ・メッセージ・フレームの検索された空のフレーム・アドレス及び長さを用いてＩＯＰシステムＤＭＡ９２０をプログラムする(ステップＡ２７)。次に、ＩＯＰシステムＤＭＡ９２０は、ＰＣＩバスのための仲裁を行い、ＰＣＩバスを獲得したときには、システムＤＭＡ９２０は、リプライ・メッセージ・フレームをシステム・メモリ９０

２の中にある空のメッセージ・フレーム・アドレスに移動させる(ステップＡ２８)。いったん１つ又は複数のリプライ・メッセージ・フレームが転送されると、インターフェース・マネジャ９１６は、メッセージ・トランスポート・マネジャ９１４に告知する(ステップＡ２９)。告知されると、メッセージ・トランスポート・マネジャ９１４は、リプライ・メッセージ・フレーム・アドレスを、Post_List循環キューの中のTail_Pointerの位置に記憶し、Tail_Pointerは、インクリメントされる(ステップＡ３０)。

メッセージ・トランスポート・マネジャ９１４は、ＩＯＰドライバ９１２に、処理するためのリプライ・メッセージ・フレームが存在することを告知する(ステップ３１)。示されている例では、このメカニズムは、レジスタ／割り込みベースの作用である。ＩＯＰドライバ９１２は、Post_List循環キューの中のHead_Pointerから、リプライ・メッセージ・フレームのアドレスを検索する(ステップ３２)。ＩＯＰドライバ９１２は、１つ又は複数のリクエスト･メッセージ・フレーム・アドレスを、システム・メモリの中に記憶することができ、そして、

リプライ・メッセージ・フレームを処理する。このオプションは、いくつかのシステムのパフォーマンスを改善させうる。ステップＡ３２の場合のように、Post_List循環キューの中のHead_Pointerがインクリメントされる。ＩＯＰドライバ９１２は、リプライ・メッセージ・フレームを検索し、それらを、ＯＳＭに送り戻す(ステップＡ３２)。ＩＯＰドライバ９１２は、現在の新たなリプライ・メッセージ・フレームのアドレスを、Free_List循環キューの中のTail_Pointerの位置に配置し、Tail_Pointerをインクリメントする(ステップＡ３４)。こうすることによって、リプライ・メッセージ・フレームは、空のメッセージ・フレーム・リソースに変換して戻される。

送信コンテキスト・マネジャ８７７は、多数の責任を有している。これには、ＴＣＢの送信キューから、プロトコル・エンジン８０２によって作成されるデータを読み出すことが含まれる。ＴＣＢは、メモリ・データ経路８４４に接続されたローカル・メモリに位置している。送信コンテキスト・マネジャ８７７は、また、ＴＣＢが存在し、送信機８５８がアイドルであるときに、送信レジスタと、

バッファと、送信ＤＭＡユニット８５４（レジスタ又はＦＩＦＯレジスタ）とをロードする。更に、送信コンテキスト・マネジャ８７７は、Ｓ／Ｇリストからの必要に応じて、送信ＤＭＡユニット８５４にＳ／Ｇエントリを与え、プロトコル・エンジン８０２への割り込みを除去する。送信コンテキスト・マネジャ８７７は、また、それぞれのＴＣＢが、ポインタをＴＣＢデータ構造のリンクされたリストに再度書き込むことによって送信を終了したときに、ＴＣＢを、送信キューからフリー・キューへ再度リンクする。

次に図１１を参照すると、本発明の好適実施例に従って送信コンテキスト・マネジャにおいて実現されているプロセスを図解する流れ図が、示されている。このプロセスは、ＩＯＰからのキックが生じたかどうかを判断することによって開始する（ステップ１１０）。「キック」は、Ｔｘコンテキスト及びレジスタ・ユニット８６８への権利（right）である。このプロセスは、ＩＯＰからのキックが生じるまで、ステップ１１００に戻り続ける。この時点で、転送キュー・ヘッダ・ポインタがレジスタから読み出される（ステップ１１０２）。その後で、第１のＴＣＢが読み出され（ステップ１１０４）、右フレーム・ヘッダが送信機に書き込ま

れる（ステップ１１０６）。次に、このフレームに対するペイロードがローカル・ペイロードであるかシステム・ペイロードであるかに関する判断がなされる（ステップ１１０８）。ペイロードがローカル・ペイロードである場合には、そのペイロードは、送信機バッファに書き込まれる（ステップ１１１０）。示されている例では、送信機バッファは、ＴｘＯバッファ８６７である。次に、シーケンスが送信されたかどうかが判断される（ステップ１１１２）。シーケンスは、フレーマ・ユニット８６５によって作成される一連のフレームであり、Ｔｘコンテキスト及びレジスタ・ユニット８６８におけるレジスタの単一のプログラミングによって開始される。プロセスは、シーケンスが送信されるまでは、ステップ１１１２に戻り続ける。シーケンスが送信されると、ＴＣＢが送信機キューから除去され（ステップ１１１４）、フリー・キュー・テール・ポインタが読み出される（ステップ１１１６）。この読み出しは、Ｔｘコンテキスト及びレジスタ・ユニット８

６８からである。ＴＣＢは、フリー・キュー・テールにリンクされる(ステップ１１１８)。次に、フリー・キュー・テール・ポインタが更新され(ステップ１１２０)、送信キュー・ヘッダ・ポインタが更新される(ステップ１１２２)。ポインタの更新は、Ｔｘコンテキスト及びレジスタ・ユニット８６８に対して右側である。そして、ＴＣＢが存在するかどうかの判断がなされる(ステップ１１２４)。別のＴＣＢが存在する場合には、プロセスは次のＴＣＢを読み出し(ステップ１１２６)、プロセスは、既に述べたように、ステップ１１０６に進む。別のＴＣＢが存在しない場合には、キューの最後に到達し、プロセスは、ステップ１１００に戻る。

再びステップ１１０８を参照すると、ペイロードがシステム・ペイロードである場合には、プロセスは、Ｓ／ＧエントリをＳ／Ｇリストから読み出す(ステップ１１２８)。次に、Ｓ／Ｇエントリが、送信Ｓ／ＧＦＩＦＯの中にロードされる(ステップ１１３０)。そして、Ｓ／Ｇエントリが更に存在するかどうかが判断される(ステップ１１３２)。これ以上Ｓ／Ｇエントリが存在しない場合には、プロセスは、既に述べたように、ステップ１１１２に進む。それ以外の場合には、プロセスは、Ｓ／ＧＦＩＦＯがいっぱいであるかどうかを判断する(ステップ１１３４)。Ｓ／ＧＦＩＦＯがいっぱいではない場合には、プロセスは、ステップ１１２８に戻る。そうでなければ、プロセスは、ＦＩＦＯがいっぱいでなくなるまで、

ステップ１１３４に戻り続ける。

受信コンテキスト・マネジャ８７８は、ファイバ・チャネルの受信コンテキスト管理に自動を提供し、それによって、他のデバイス及び／又はプロトコル・エンジン８０２などのシステム・リソースのワークロードを軽減する。受信コンテキスト・マネジャ８７８は、コンテキスト管理、ＤＭＡ開始、ＤＭＡ更新、フレーム完成（complete）処理など、様々な機能を提供する。コンテキスト管理には、ファイバ・チャネルのヘッダ情報を受信制御ブロック（ＲＣＢ）に調停（reconcile）することを含むが、これは、ファイバ・チャネル・シーケンス情報を有効化しデータ転送パラメータを特定する手段を提供する。ＤＭＡ開始機能は、フ

ァイバ・チャネル・ヘッダ・パラメータ・フィールドを正しいバッファ・オフセットにマップしＤＭＡ転送を開始することによって、初期開始点を用いて受信ＤＭＡ８５６をプログラムすることを含む。ＤＭＡ更新機能には、フレームＤＭＡ転送を維持するのに要求される追加的なバッファ・アドレス／長情報を用いて、受信ＤＭＡ８５６を更新することが含まれる。フレーム完成処理では、受信ＤＭＡ情報の更新と、ファイバ・チャネル・シーケンス完成の検出と、条件的な完成報告とが、生じる。

次に、図１２を参照すると、本発明の好適実施例による受信された制御ブロックのためのフォーマットが示されている。受信制御ブロック１２００は、ファイバ・チャネル・シーケンスを管理するのに要求される情報を含む。ファイバ・チャネル・ヘッダ・フィールド１２０２は、入来フレームを有効化するのに用いられる情報を含む。シーケンス状態情報フィールド１２０４は、ファイバ・チャネル・シーケンスをトラッキングし管理するのに用いられる。ＤＭＡ情報フィールド１２０６は、ファイバ・チャネル・データの宛先アドレスへのマップをトラッキングし管理するのに用いられる。最後に、タイム・スタンプ・フィールド１２０８は、シーケンスが完成したことを指示するのに用いられる。

次に、図１３を参照すると、本発明の好適実施例によるコンテキストのスイッチングを実行するのに用いられる流れ図が示されている。コンテキストのスイッチングは、フレーム受信機が受信されたコンテキスト・マネジャにコンテキストのスイッチングが必要であることを告知するときに開始する。この状況は、現在

のファイバ・チャネル・ヘッダが現在確立している受信コンテキストと一致しないときに生じる。この時点で、受信コンテキスト・マネジャは、ＲＣＢの中にある次のコンテキストを見出す(ステップ１３００)。次に、そのフレームがこのＲＣＢに対する新たなファイバ・チャネル・シーケンスの第１のフレームであるかどうかに関する判断がなされる(ステップ１３０２)。受信されたフレームが特定のシーケンスに対して受信された第１のフレームであるかどうかという判断は、ＲＣＢのコンテキスト状態ワードの中の「アクティブ」ビットに基づく。このビットは、当初、プロトコル・エンジンによって０に設定され、この特定のシーケ

ンスがアクティブではない(フレームがまだ受信されていない)ことを指示する。受信コンテキスト・マネジャは、コンテキストのルックアップ及びスイッチングを実行し、関連するＲＣＢがまだこのビット・セットを有していないと判断するときには、ステップ１３０４に説明されている作用を実行し、ＲＣＢのアクティブ・ビットを１に設定する。回答が肯定である場合には、ＲＣＢは、シーケンスＩＤ（Ｓ＿ＩＤ）と受信ＩＤ（ＲＸ＿ＩＤ）とを用いて更新され、シーケンス・カウント（ＳＥＱ＿ＣＮＴ）を用いて、このシーケンスに対して予測される値が設定される(ステップ１３０４)。

次に、フレーム受信コンテキストが有効であるかどうかが判断される(ステップ１３０６)。フレーム受信コンテキストは、先行するシーケンスが不完全である場合には、典型的には有効である。ステップ１３０６は、先行する完全なシーケンスの状態を記憶し、それによって、シーケンスは、別のフレームがそのシーケンスに対して受信されるときには後の時点で完成されることになる。この判断は、また、第１のフレームがこのＲＣＢ出ない場合には、ステップ１３０２から直接になされる。フレーム受信コンテキストが有効である場合には、先のフレーム受信コンテキストは、メモリにセーブされ(ステップ１３０８)、新たなワレーム受信コンテキストがメモリからロードされる(ステップ１３１０)。フレーム受信コンテキストが有効でない場合には、プロセスはステップ１３０８をスキップして、メモリからの新たなフレーム受信コンテキストをロードするステップ１３１０に進む。次に、信号が受信機に送られ、フレーム受信コンテキストを再評価する(ステップ１３１２)。プロセスは、その後で終了する。

次に、図１４を参照すると、本発明の好適実施例によるＤＭＡ開始プロセスの流れ図が示されている。フレーム受信機がファイバ・チャネル・ヘッダを評価し、このヘッダが先にロードされ現在確立している受信コンテキストと一致していると判断するときには、フレーム受信機は、受信コンテキスト・マネジャに、フレーム転送が必要であることを告知する。この時点で、受信コンテキスト・マネジャは、ＤＭＡ開始プロセスを開始させる。このプロセスは、フレーム相対オフセットがＲＣＢの現在相対オフセットよりも大きいかどうかを判断することによ

って開始する(ステップ１４００)。この判断への回答が否定であるときには、プロセスは、正しいエントリが見つかるまでＳ／Ｇリストを逆方向に走査する(ステップ１４０２)。回答が肯定である場合には、正しいエントリが見つかるまでＳ／Ｇリストを順方向に走査する(ステップ１４０４)。ステップ１４００ないし１４０４は、ファイバ・チャネルのヘッダ・パラメータ・フィールドをＲＣＢの現在の相対オフセット・フィールドと比較し、ＲＣＢのベースＳ／Ｇポインタとの現在のＳ／Ｇポインタとを用いてＤＭＡＳ／Ｇリストを順方向又は逆方向に走査し、正しい開始Ｓ／Ｇ要素を見つけることによって、適切な開始ＤＭＡＳ／Ｇ要素（ＲＣＢ現在のＳ／Ｇポインタ）を見つけるのに用いられる。これらのステップにより、順不同（out of order）で受信されたフレームが正しいＤＭＡバッファ・アドレスに適切にマップされることが可能となる。

次に、アドレス及び長さの情報が調整される(ステップ１４０６)。このステップは、見出されたＳ／Ｇ要素内のこのフレームに対する実際の開始及び長さを、ファイバ・チャネルのヘッダ・パラメータ・フィールド及びＲＣＢの現在のＳ／Ｇポインタ・アドレスと、ＲＴＣＢの現在のＳ／Ｇポインタの長さと、ＲＣＢの現在の相対オフセットとに基づいて、計算する。次に、受信ＤＭＡは、アドレス及び長さ情報を用いてプログラムされ、ＤＭＡ転送が開始される(ステップ１４０８)。プロセスは、その後で終了する。

次に図１５を参照すると、本発明の好適実施例によるＤＭＡ更新プロセスの流れ図が示されている。ファイバ・チャネルのフレーム・ペイロード・データが複数のＤＭＡＳ／Ｇエントリにわたるときには、受信コンテキスト・マネジャは、追加的なＤＭＡプログラミング情報を受信ＤＭＡユニットに提供し、それによって、フレームＤＭＡ転送が継続することになる。受信ＤＭＡユニットは、追加的なＳ／Ｇエントリの必要性を告知するが、これにより、次のＳ／Ｇ要素を取得しこのＳ／Ｇ要素に関係する現在の相対オフセットを更新することによって(ステップ１５００)、プロセスが開始する。その後で、受信ＤＭＡユニットは、ステップ１５００において決定されたアドレス及び長さ情報を用いてプログラムされ、ＤＭＡ転送は、データの転送を継続を開始する(ステップ１５０２)。プロセス

は、この後で終了する。

次に、図１６を参照すると、本発明の好適実施例によるフレーム完成処理プロセスが示されている。受信ＤＭＡユニットからの信号のフレーム転送によって指示されているように、受信ＤＭＡ動作の完了の際に、受信コンテキスト・マネジャは、ＲＣＢにおけるＤＭＡ転送情報を更新することによって、フレーム完成処理を開始する(ステップ１６００)。その後で、シーケンスの最後に到達したかどうかに関する判断がなされる(ステップ１６０２)。シーケンスの最後に到達した場合には、タイマが読み出され、ＲＣＢタイム・スタンプ・フィールドが、ＥＭＢレジスタ及びタイマ・ユニット８５０からタイマ値と共に書き込まれる(ステップ１６０４)。その後で、完成報告が禁止されているかどうかが判断される(ステップ１６０６)。報告が禁止されていない場合には、ＲＣＢポインタが、シーケンス完成キューに書き込まれる(ステップ１６０８)。プロセスは、その後で終了する。

再びステップ１６０６を参照すると、報告が禁止されるべき場合には、プロセスは、終了する。プロセスは、また、ステップ１６０２においてシーケンスの最後に到達した場合にも終了する。

リンク制御ユニット８６４は、２つのポートの間のリンクである。リンク制御ユニット８６４は、チップ８００内でデータの送受信をする際に用いられる。リンク制御ユニット８６４によって提供される機能は、リンクの幾何学的態様（トポロジ）に依存する。例えば、リンクがポイント間のものである場合には、リンク制御ユニット８６４は、データの転送を提供するだけである。仲裁されたループ・トポロジが用いられている場合には、リンク制御ユニット８６４は、ループの管理する追加的な機能を提供する。「ループ」という用語は、一方向的にデータ

を転送するように接続された集合ノードを意味する。それぞれのノードは、ループ上のデータに対するソース又は宛先であり、例えば、アダプタ、コンピュータ、遠隔ストレージ・ユニットなどでありうる。

次に図１７を参照すると、本発明の好適実施例によるループ管理制御のための

ステート・マシンが示されている。転送制御ユニット８８０の内部に実現されるステート・マシン１７００は、リンク制御ユニット８６４である。リンク制御ユニット８６４は、示されている例では、仲裁されたループへのアクセスを提供している。仲裁されたループでは、仲裁プロセスは、どのノードが送信データへの権利を有しているのかを判断するのに用いられる。リンク制御ユニット８８０は、ループに対するリクエストを送出することによってデータ転送のための仲裁されたループへのアクセスを得ようと試みる。リクエストは、ＡＲＢプリミティブとも称される仲裁プリミティブを送ることによってなされる。ループが取得されると、本発明は、ノード及びループのアクティビティをモニタして、ループの全体的なパフォーマンスを最大化する。本発明は、トラフィックがすぐに準備が完了するのか、停止するのかを識別して評価し、更に、ループのパフォーマンスを最大化する際にループ上でどのようなアクティビティが要求されているのかを識別し、評価する。

ステート・マシン１７００は、アイドル状態であるステートＳ１で始まり、特定の宛先亜又はノードに対してループを取得するというリクエストがなされるまで、この状態にとどまる。ループが取得されるべきときには、状態Ｓ１にあるステート・マシン１７００は、ループに対し、仲裁プロセスに入ってループの所有権を取得するように求める。ループは、データがターゲット又は宛先ノードへの送信のために読み出されるときにだけ、又は、宛先ノードへ送信されるデータが入手可能であることがわかるとすぐにだけ、要求される。

ループを取得することを求めるリクエストに応答して、ステート・マシン１７００は、ループ状態を待機する状態Ｓ２にシフトし、そこで、ステート・マシン１７００はループの所有権が得られた後で開放（open）プリミティブを送出する。状態Ｓ２では、ステート・マシン１７００は、ループが、ステート・マシン１７００が実行しているノードによるデータ転送のために使用可能となるのを待機す

る。ノードは、ループを求めて仲裁を行いループを獲得したときには、仲裁獲得状態にある。「開放」（ＯＰＮ）プリミティブは、そのプリミティブがターゲット

・ノードを特定するループの上に送られる。ノードは、ＯＰＮプリミティブが送られるときには、「開放」ノードであると考えられる。開放ノードは、ソース・ノードである。開放ノードは、ループを求める仲裁プロセスに再度入ることを要求されることなく、異なるノードへの接続を断絶させたり確立したりする。「開放された」ノードは、開放ノードが接続を確立しているノードである。開放されたノードは、宛先又はターゲット・ノードである。開放されたノードは、どれかが入手可能であるならば、データを開放ノードに戻すことができるが、他のどのノードにもデータを送ることは許されない。「接続」は、ソース・ノードがデータを送ることを希望している宛先ノードを識別する開放ノードによってなされる。接続は、ＯＰＮプリミティブが送られるときに確立される。接続を確立するには、完全なハンドシェークは要求されない。接続の閉鎖だけが、完全なハンドシェーク、すなわち、プリミティブの変換を必要とする。

ステート・マシン１７００は、接続が要求されたノードに対し確立されることに応答して、オン・ループ状態である状態Ｓ３にシフトし、規則によって、フレームの送出が可能となる。「フレーム」とは、ヘッダ情報が追加されたデータのパケットである。一般に、すべてのデータがオン・デマンドで入手可能であることがわかるまではフレームを送り始めることは不可能である。その理由は、フレームの送信は、示されている例では、中断することができないからである。ステート・マシンが状態Ｓ３にあるときには、フレームは、ループ上をノードまで送ることが可能である。フレームの完成に応答して、ステート・マシン１７００は、判断ウィンドウ状態である状態Ｓ４にシフトして、追加的なフレームが送る準備ができているかどうかが検討される。また、追加的なフレームが先のフレームと同じノードに送られる準備ができており、更に、宛先が追加的なフレームを受け入れる準備ができている場合には、ステート・マシン１７００は、状態Ｓ３に戻る。ハンドシェーキングのプロセスが、送信側のノードと受信側のノードとの間で生じ、受信側ノードにおけるバッファがオーバーランすることを防止する。このハンドシェーキングはまた、「クレジット」とも称される。状態Ｓ３と状態Ｓ４

との間のこのシフトは、データのフレームが宛先に向けて転送される準備ができている限りなされる。状態Ｓ４では、ノードに送るための追加的なデータが入手可能ではない、又は、規則がループの開放を命じている場合には、ステート・マシン１７００は、閉鎖状態の待機である状態Ｓ５にシフトし、それ以上のデータはノードに送られない。この状態では、「閉鎖」（ＣＬＳ）プリミティブがループ上に送られ、ループを開放する。閉鎖プリミティブのハンドシェーキングが完了しループがもはや開放ではなくなると、ステート・マシン１７００は、状態Ｓ１のアイドル状態に戻る。閉鎖が生じると、ループの所有権（ownership）は必ずしも放棄されるとは限らない。変化が、decision_window-waiting_for_close_idleから生じると、ループの所有権は放棄される。所有権の放棄を望まない場合には、decision_window-waiting_for_transfer-waiting_for_openループが選択される。これは、基本的には、先のルートと同じＣＬＳハンドシェーキングを行うが、ループの所有権は保持される。これは、ＯＰＮをwaiting_for_open状態では直ちに送ることが許されないことの理由の一部である。

再び状態Ｓ４を参照すると、データの追加的なフレームが入手可能であり許容されてはいるが、異なるノードに対するものであって異なるノードが開放されることを要求している場合には、ステート・マシン１７００は、転送状態の待機である状態Ｓ６にシフトする。状態Ｓ６では、ステート・マシン１７００は、別のノードへの転送リクエストを送る。状態Ｓ６におけるこのリクエストは、閉鎖プリミティブの送出と、戻されるべき閉鎖プリミティブの待機とを含む。先の接続が閉鎖されたと判断するが、規則によって開放を送ることが許されない場合には、ステート・マシン１７００は、開放状態の待機である状態Ｓ７にシフトし、この状態において、ステート・マシン１７００は、開放プリミティブを送り、新たなノードを開放する。状態Ｓ７では、ステート・マシン１７００は、データ転送の前にタイム・ギャップが生じるように、待機する。要求されたノードへの接続が確立されると、ステート・マシン１７００は、状態Ｓ３にシフトして、データのフレームをそのノードに送る。

状態Ｓ６に戻ると、閉鎖プリミティブへの応答を待機している間に、ステート

マシン１７００は、新たなノードが廃棄されるようにとのリクエストを要求する規則に応答して、状態Ｓ５にシフトし、ループを閉鎖する。状態Ｓ１では、閉鎖プリミティブが受信され、閉鎖プリミティブを伴う応答が要求されると、ステート・マシンは状態Ｓ５にシフトして閉鎖プリミティブを送り、状態Ｓ１に戻る。再び状態Ｓ２を参照すると、ループへのアクセスを待機している間に、ステート・マシン１７００は、応答における閉鎖プリミティブを求める閉鎖プリミティブを受信することに応答して、又は、送信リクエストが廃棄されることを規則が求めている場合には、状態Ｓ５にシフトする。

次に図１８を参照すると、本発明の好適実施例による開放状態におけるループ管理のためのプロセスの流れ図が示されている。このプロセスは、あるノードが、ステート・マシンが動作しているノードを開放したかどうかと、クレジットの不足のためにデータ送信が行われていないかどうかと、合理的な時間周期の後で接続を閉鎖していないかどうか（ステップ１８００）を判断することによって開始する。データ転送がなされないままで選択された時間周期が経過した場合には、プロセスは、接続を閉鎖し(ステップ１８０２)、その後でプロセスが終了する。そのような状況が存在しない場合には、データ送信のリクエストが除去されたかどうかに関する判断がなされる(ステップ１８０４)。リクエストは、様々な理由で除去されうる。例えば、ホストからのリクエストや、エラー条件の回復、また、送信機がデータ送信のリクエストを除去することもありうる。リクエストが除去されると、接続は閉鎖され(ステップ１８０６)、ループの所有権は開放され(ステップ１８０８)、プロセスは、その後で終了する。

次の図１９を参照すると、本発明の好適実施例によって、アイドル状態でループの取得を制御するのに用いられる規則を組み入れているプロセスの流れ図が示されている。示されている例では、ポートは「欲張りな」(greedy)状態におかれており、この状態では、いったん得られたループの所有権は、データが再び入手可能となる可能性が存在する限り、保持される。更に、この状態では、ノードは、別のノードがループへのアクセスを望んでいることを検出しない場合には、ループの所有権を保持する。

プロセスは、欲張りな状態が存在するかどうかを判断することによって開始す

る(ステップ１９００)。欲張りな状態が存在する場合には、プロセスは、ノードがデータをロードしているかどうかを判断する(ステップ１９０２)。このノードは、ソース・ノードである。ノードがデータをロードしていない場合には、プロセスは、ステート・マシンをアイドル状態に維持する(ステップ１９０４)。プロセスは、その後に終了する。ノードがデータをロードしている場合には、ループが取得され(ステップ１９０６)、ステート・マシンが、図１７に図解されているように、ループ待機状態にシフトされる(ステップ１９０８)。プロセスは、その後で終了する。再びステップ１９００を参照すると、欲張りの状態が存在していない場合には、プロセスは、完全なフレームが送られる準備ができているかどうかを判断する。完全なフレームが送られる準備ができている場合には、プロセスは、ライブ（live）を取得するように進み(ステップ１９０６)、ループ待機状態にシフトされる(ステップ１９０８)。プロセスは、その後で終了する。完全なフレームを送る準備ができていない場合には、ステート・マシンは、アイドル状態に維持され(ステップ１９１２)、プロセスは、後で終了する。図１９は、複数のステップがアイドル状態の間に反復される単一のパスを通過するプロセスを示している。

図２０を参照すると、本発明の好適実施例に従ってループ待機状態において用いられる規則を組み入れているプロセスの流れ図が図解されている。このプロセスは、ノードがデータ転送のための別のノードによって開放された場合に用いられる。プロセスは、ループを取得しようとし(ステップ２０００)、他方で、データを遠隔ノードに送ることを試み、このプロセスが実行されるノードを開放する(ステップ２００２)。ループが取得されたかどうかに関する判断がなされる(ステップ２００４)。ノードがまだ取得されていない場合には、プロセスは、ステップ２０００に戻る。ループ取得の際には、プロセスは、ステップ２００２においてデータを送るパラレル・プロセスを含めて、終了する。また、遠隔ノードに送るために追加的データが存在するかどうかを判断する(ステップ２００６)。データが依然として入手可能である場合には、プロセスは、ステップ２００２に戻る。送られるべき追加的データが存在しない場合には、プロセスは、ステップ２０００におけるループ取得の試みの終了も含めて、終了する。このプロセスは、

デー

タ転送のために別のノードによってそのノードが開放されるかどうかを判断する。ただし、この遠隔ノードがループが取得されたときに転送されることが予定されているデータのための宛先ノードである。ノードが宛先ノードでもある別のノードによって開放された場合には、データは、好ましくは、この遠隔ノードに送られる。そして、更に多くのデータが送られるために存在しているかどうかが判断される（２００４）。送られるデータが更に存在するならば、ステート・マシンは、ループを取得する（ステップ２００８）。プロセスは、その後で終了する。再びステップ２００４を参照すると、送るべきデータがそれ以上存在しない場合には、ステート・マシンは、閉鎖待機状態にシフトされる（ステップ２００８）。プロセスは、その後で終了する。再びステップ２０００を参照すると、ノードが遠隔ノードによって開放されていない場合には、プロセスは、ステップ２００６においてループを取得するように進む。これらのステップはシリアルであるように示されているが、プロセスは、実際には、２つのパラレル・プロセスとして生じる。連続的にループを取得する間に、データは、可能であれば、送られるのが好ましい。これらのプロセスは、すべてのデータが送られるまで、又は、ループが要求されるまで、パラレルに継続する。

次に図２１では、本発明の好適実施例に従って判断ウィンドウ状態における変化（transitions）を扱うのに用いられる規則を組み入れているプロセスの流れ図が示されている。プロセスは、ループに対するリクエストが別のノードによってなされたかどうかを判断することによって開始する（ステップ２１００）。ループが別のノードによって要求されていない場合には、クレジットの不足のためにデータが送信されないかどうか、更に、データが受信されていないかどうかが判断される（ステップ２１０２）。このような条件が存在しない場合には、ループが保持されていた時間の長さをトラッキングするタイマが、リセットされる（ステップ２１０４）。プロセスは、その後で終了する。ステップ２１０２を参照すると、データが送信又は受信されていない場合には、プロセスは、ループが保持されていた時間が選択された時間よりも長いかどうかがを判断する（ステップ２１

０８）。この選択された時間周期は、実現例に依存してプログラム可能である。ループが保持されていた時間が選択された時間周期よりも長い場合には、ステート・マシンは、閉鎖待機状態にシフトされ(ステップ２１０８)、プロセスは、その後で終了する。そうでない場合にも、プロセスは、停止する。ステップ２１００を再び参照すると、ループに対するリクエストが別のノードによってなされ、プロセスは、また、既に述べたように、ステップ２１０６に進む。

次に図２２を参照すると、本発明の好適実施例に従って判断ウィンドウ状態における変化を処理する規則を組み入れているプロセスの流れ図が示されている。このプロセスは、ノードが公平（フェア、fair）な状態である場合に実行される。換言すると、ノードは、残りのノードがループを要求するときには常にそれらの残りのノードがループにアクセスすることを可能にすることを試みる。このプロセスは、ループがリクエストされているかどうかを判断することによって開始する。そうでない場合には、ステート・マシンは、判断ウィンドウ状態から閉鎖状態にシフトして、接続を閉鎖し、ループを開放する(ステップ２２０２)。その後で、ステート・マシンは、アイドル状態においてループを要求することを試みるように促される(ステップ２２０４)。プロセスは、その後で終了する。

図２３を参照すると、本発明の好適実施例に従って判断ウィンドウ状態からの変化を扱う際に用いる規則を組み入れたプロセスの流れ図が示されている。このプロセスは、ループ上のノードをポーリングする（poll）のに用いられる。このプロセスは、ノードを開放することによって開始する(ステップ２３００)。次に、プロセスは、ノードが応答する時間周期の間待機する(ステップ２３０２)。次に、追加的なフレームがどのようなものであれ受信されたかどうかが判断される(ステップ２３０４)。追加的なフレームが受信されている場合には、プロセスは、ステップ２３０２に戻る。そうでない場合には、ノードは、閉鎖される(ステップ２３０６)。次に、追加的なノードがポーリングのために存在するかどうかが判断される(ステップ２３０８)。追加的なノードが存在する場合には、プロセスは、ステップ２３００に戻り、別のノードをポーリングする。そうでない場合には、プロセスは、終了する。

以上において、本発明は、完全に機能するデータ処理システムのコンテキストに関して説明してきたが、この分野の当業者であれば、本発明のプロセスは、命令のコンピュータ可読な媒体の形式や様々な形態で提供することが可能であり、

本発明は、提供されたものを実行するのに実際に用いられる媒体を生じる特定のタイプの信号とは無関係に応用されうることが重要である。コンピュータ可読な舞いたいには、フロッピィ・ディスク、ハードディスク・ドライブ、ＲＡＭ、ＣＤ－ＲＯＭ、出る及びアナログ通信リンクなどの伝送タイプの媒体などが含まれる。

本発明の説明は、例示と説明を目的としてなされたが、網羅的なものではなく、ここで開示された形式に限定されるものでもない。当業者には、多くの修正や改変が明らかである。実施例は、当業者が考慮されている特定の使用に対して適した様々な修正を含んだ形で本発明を理解することを可能ならしめるように、本発明の原理を最良に説明するように選択され説明されてる。

【図１】

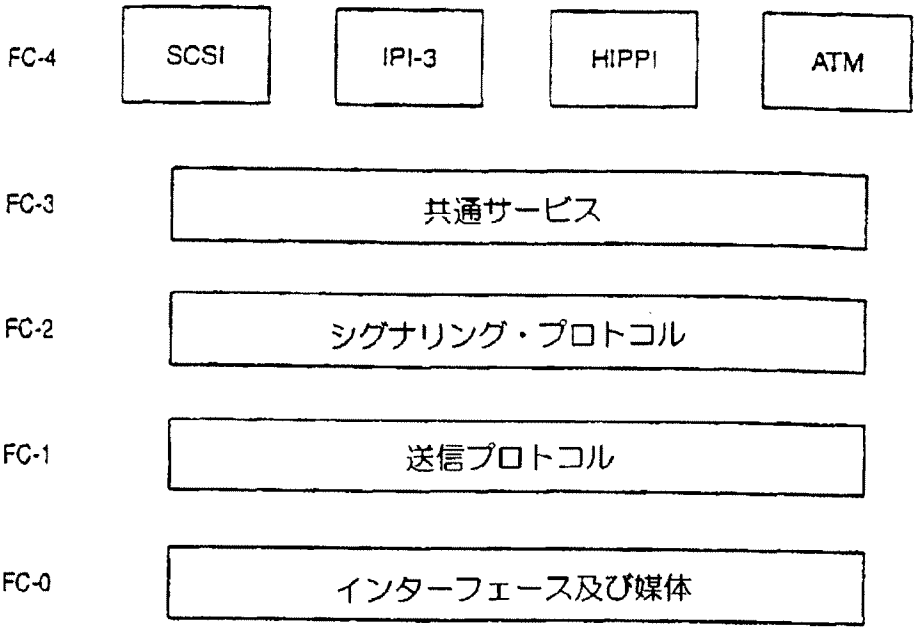


FIG. 1

【図２】

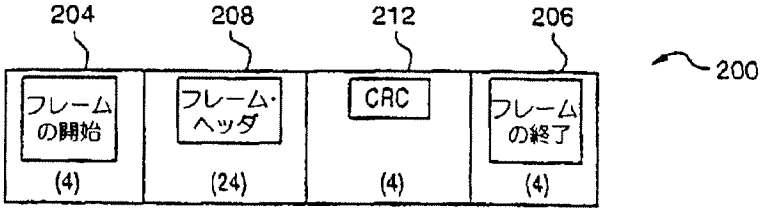


FIG. 2A

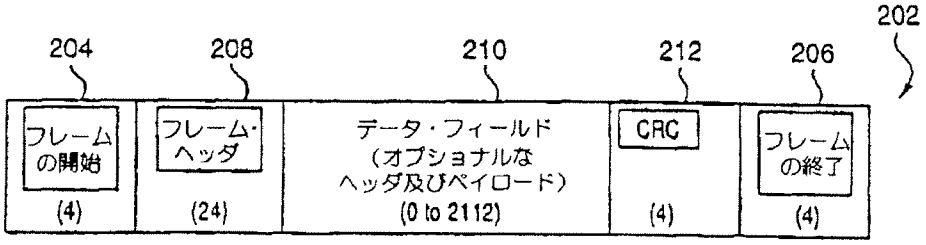


FIG. 2B

【図３】

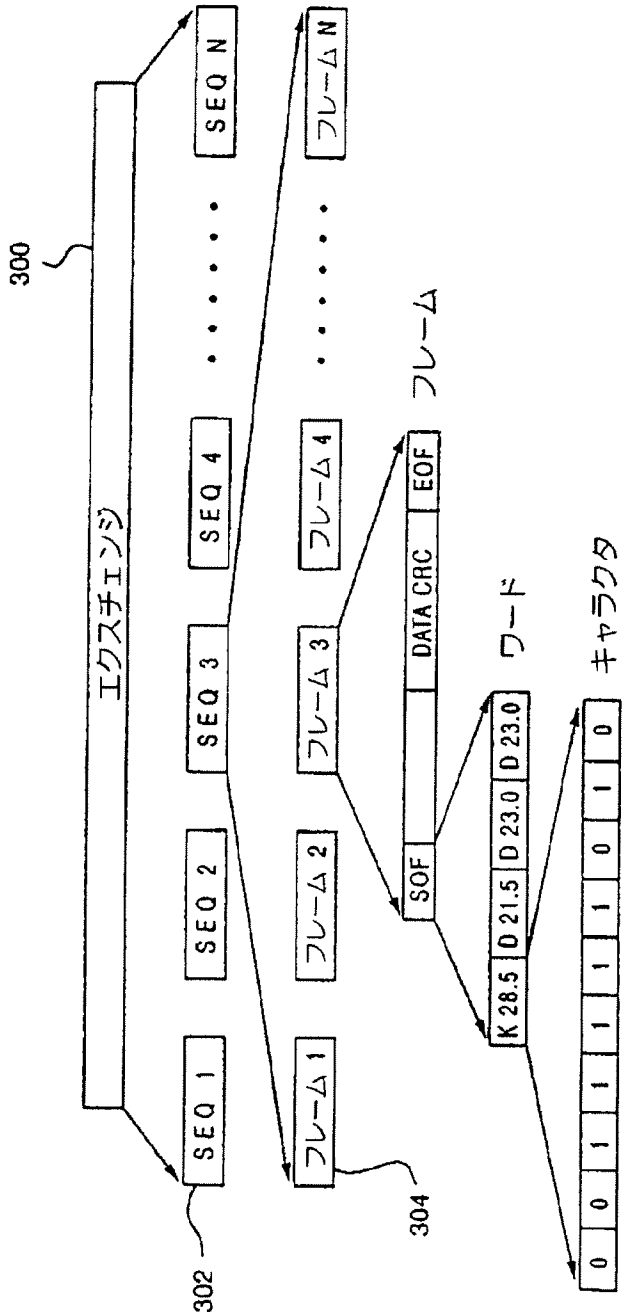

300
エクスチェンジ
SEQ 1
SEQ 2
SEQ 3
SEQ 4
SEQ N
フレーム 1
フレーム 2
フレーム 3
フレーム 4
フレーム N
302
304
フレーム
SOF
DATA CRC
EOF
ワード
K 28.5
D 21.5
D 23.0
D 23.0
D 23.0
キャラクタ
0 0 1 1 1 1 1 1 0 1 0

FIG. 3

【図４】

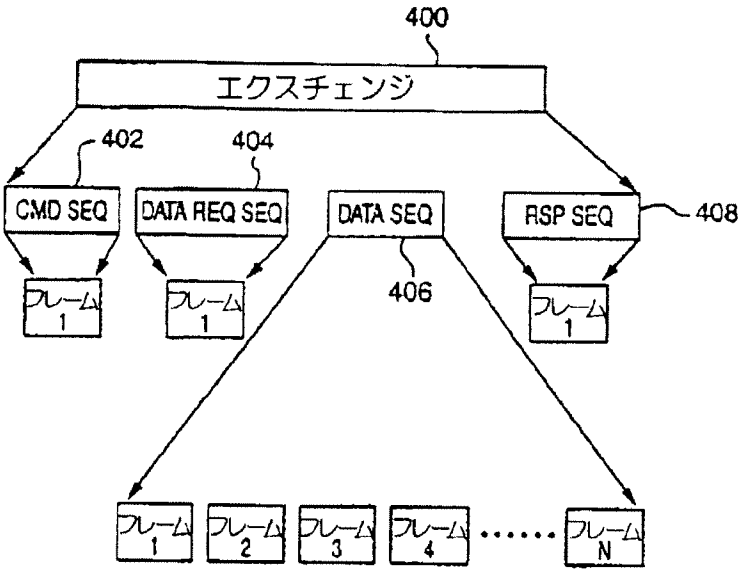


FIG. 4

【図５】

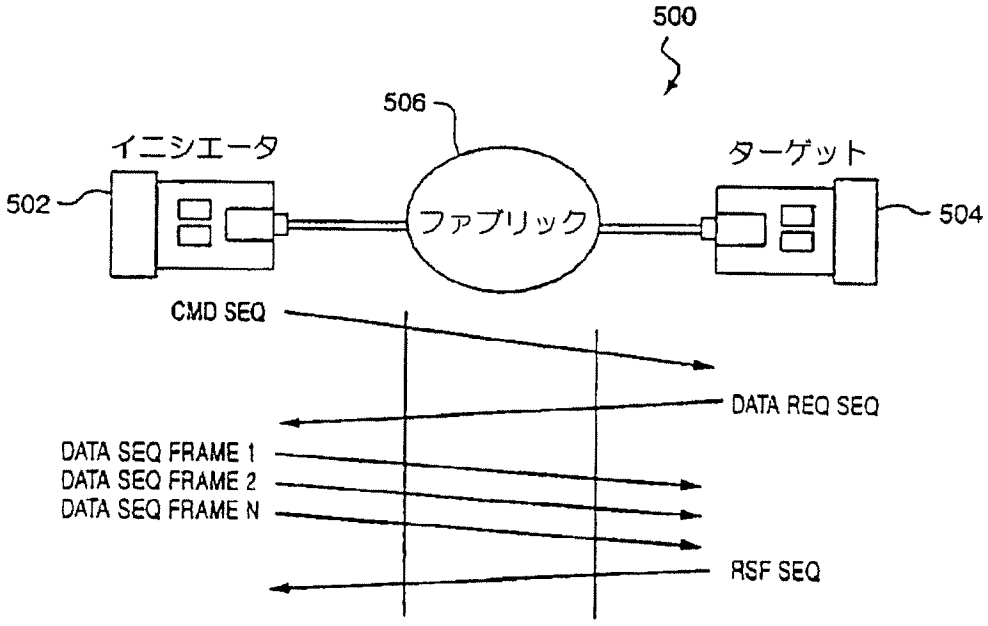


FIG. 5

【図６】

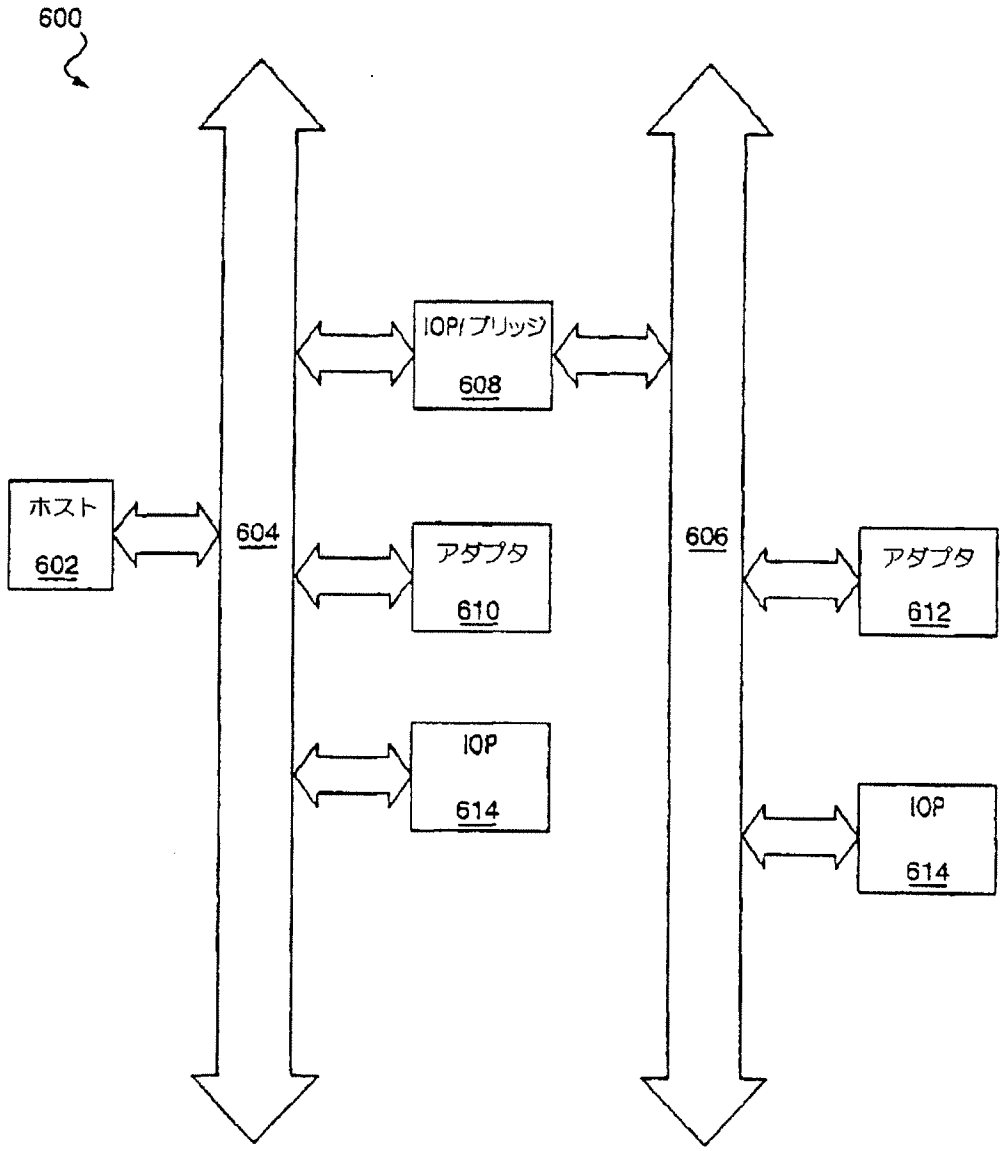
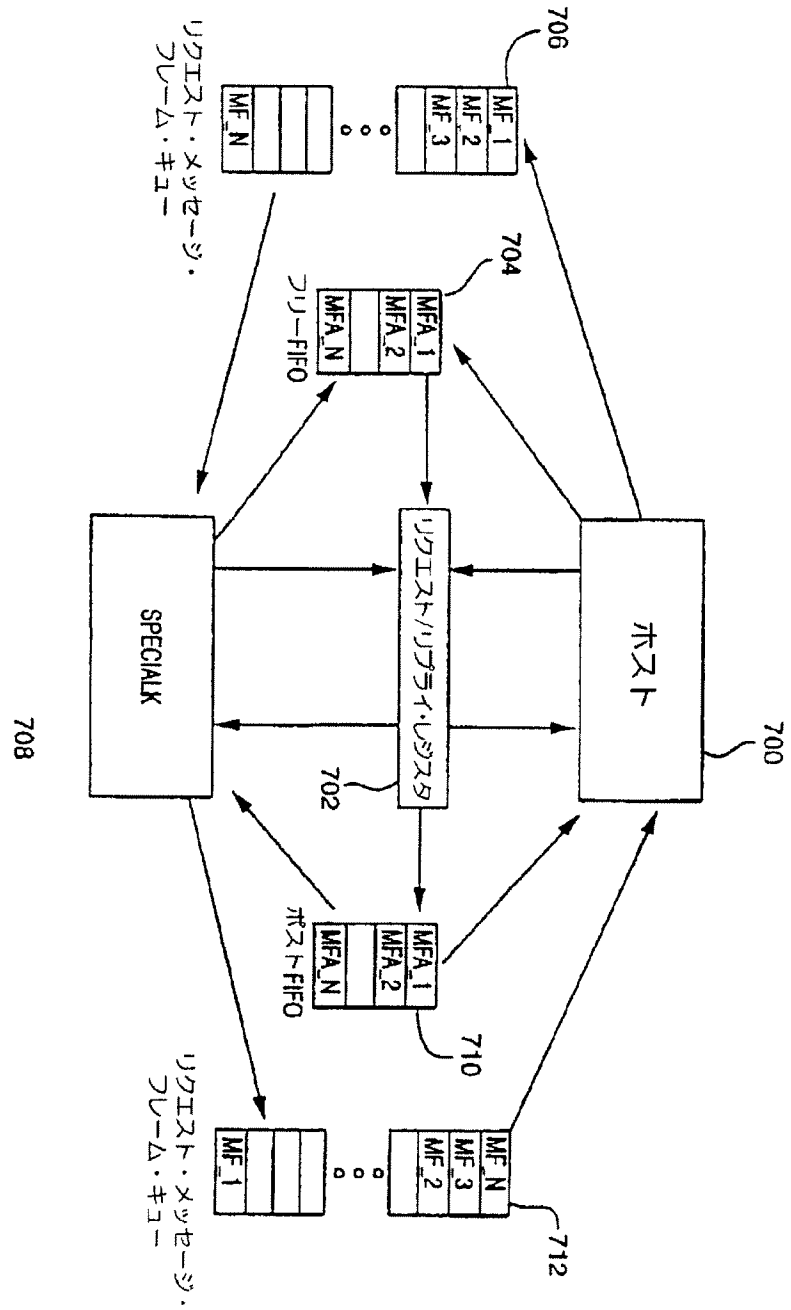

600
IOP/ブリッジ
608
ホスト
602
604
606
アダプタ
610
アダプタ
612
IOP
614
IOP
614

FIG. 6

【図７】

FIG. 7


706
MF_1
MF_2
MF_3
MF_N
リクエスト・メッセージ・フレーム・キュー
704
MFA_1
MFA_2
MFA_N
フリーFIFO
SPECIALK
リクエスト/リプライ・レジスタ
ホスト
700
702
708
ポストFIFO
710
MF_N
MF_3
MF_2
MF_1
712
リクエスト・メッセージ・フレーム・キュー

【図８Ａ】

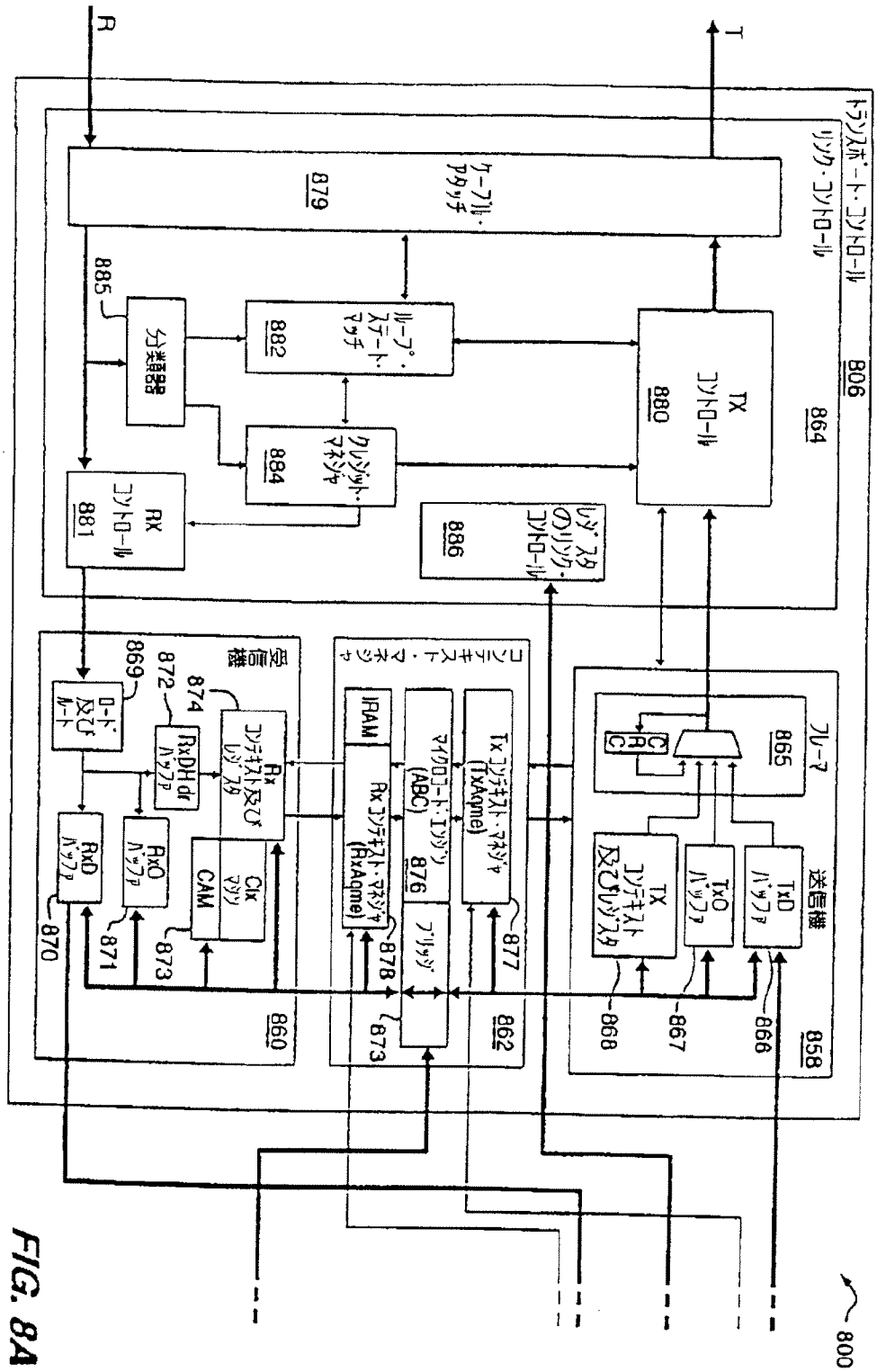
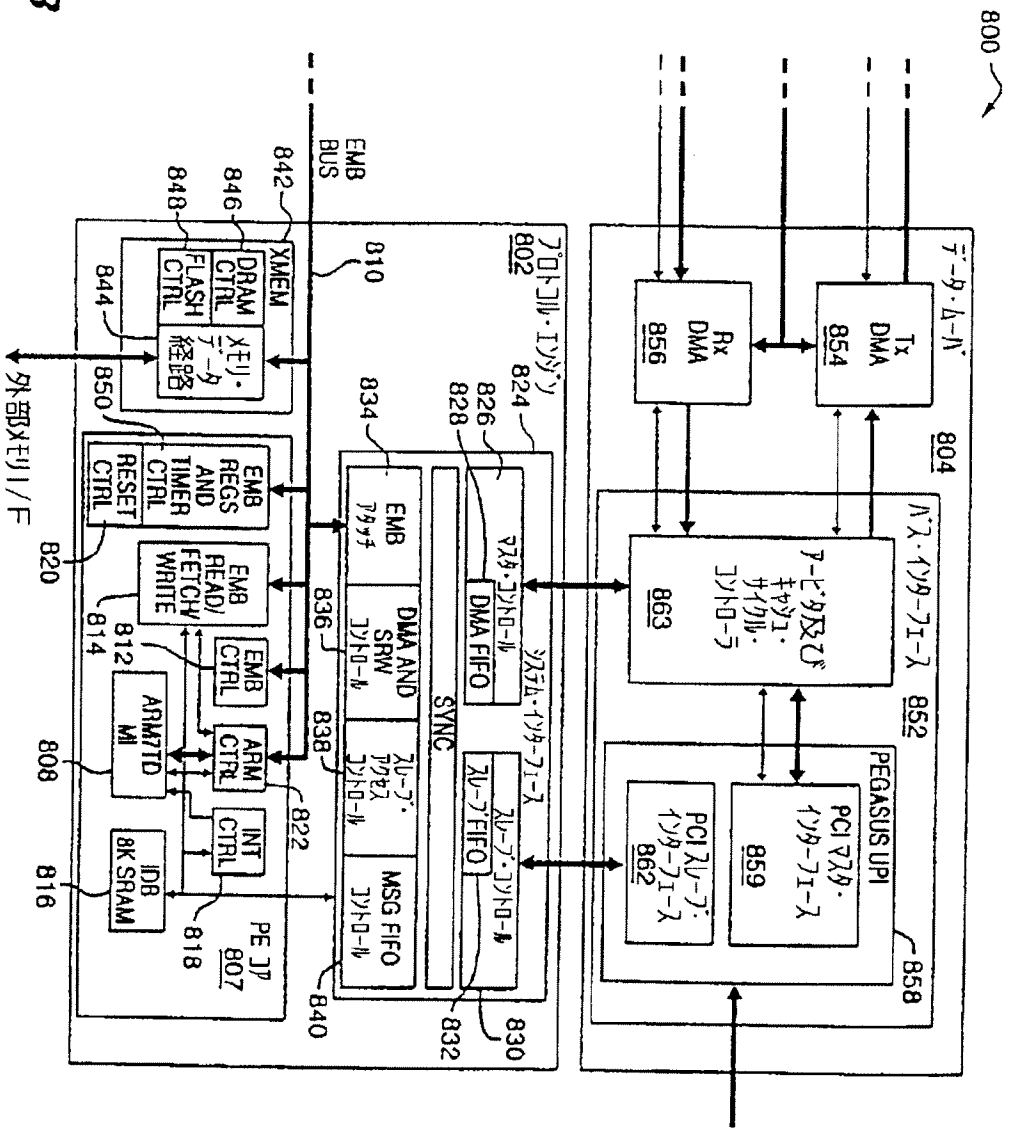

R
T
トランスポート・コントロール
リンク・コントロール
806
ケーブル・アタッチ
879
885
分類器
ループ・ステート・マッチ
882
TX コントロール
880
864
クレジット・マネジャ
884
RX コントロール
881
レジスタのリンク・コントロール
886
フレーマ
865
送信機
858
TxD バッファ
TxO バッファ
TX コンテキスト及びレジスタ
866
867
868
コンテキスト・マネジャ
IRAM
Tx コンテキスト・マネジャ (TxAqme)
マイクロコード・エンジン (ABC)
876
Rx コンテキスト・マネジャ (RxAqme)
877
878
873
862
受信機
869
872
874
ロード及びルート
RxDH dr バッファ
Rx コンテキスト及びレジスタ
RxD バッファ
RxO バッファ
Clx マジン CAM
870
871
873
860
800

FIG. 8A

【図８Ｂ】

FIG. 8B


800
804
データ・ムーバ
Tx DMA 854
Rx DMA 856
バス・インターフェース 852
アービタ及びキャッシュ・サイクル・コントローラ 863
PEGASUS UPI
PCI マスタ・インターフェース 859
PCI スレーブ・インターフェース 862
858
プロトコル・エンジン 802
824
826
828
マスタ・コントロール
DMA FIFO
システム・インターフェース
スレーブ・コントロール
スレーブ FIFO
SYNC
830
832
834
EMB アタッチ
DMA AND SRW コントロール
スレーブ・アクセス・コントロール
MSG FIFO コントロール
836
838
840
EMB BUS
842
846
848
XMEM
DRAM CTRL
FLASH CTRL
メモリ・データ経路
844
850
外部メモリI/F
810
EMB REGS AND TIMER CTRL
RESET CTRL
820
EMB READ/ FETCH/ WRITE
812
814
EMB CTRL
ARM CTRL
822
ARM7TD MI
808
INT CTRL
IDB 8K SRAM
816
818
PE コア 807

【図９】

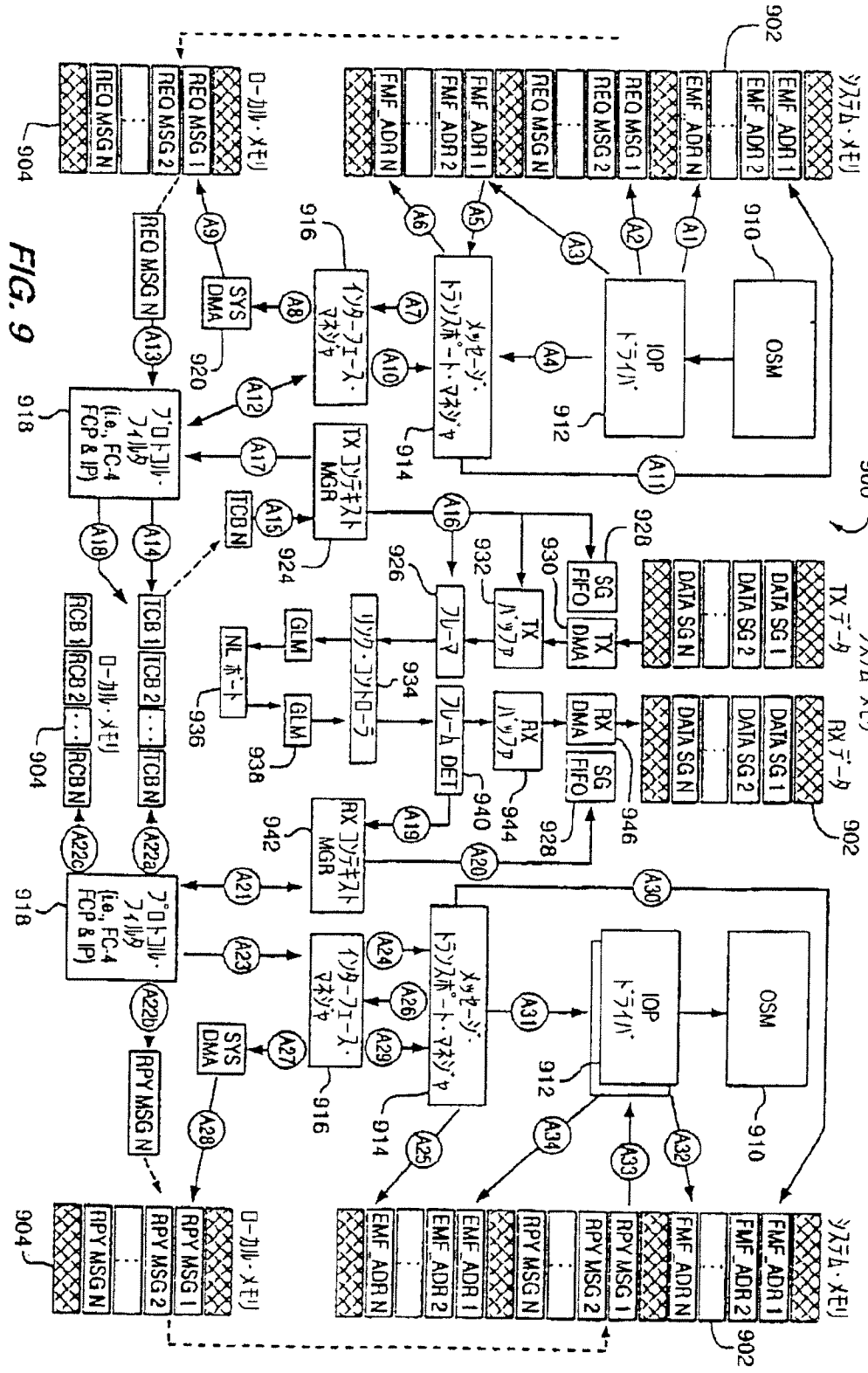

902
システム・メモリ
EMF_ADR 1
EMF_ADR 2
EMF_ADR N
REQ MSG 1
REQ MSG 2
REQ MSG N
FMF_ADR 1
FMF_ADR 2
FMF_ADR N
904
ローカル・メモリ
910
OSM
912
IOP ドライバ
914
メッセージ・トランスポート・マネジャ
916
インターフェース・マネジャ
918
プロトコル・フィルタ (i.e., FC-4 FCP & IP)
920
SYS DMA
900
928
TX データ
DATA SG 1
DATA SG 2
DATA SG N
SG FIFO
TX DMA
930
932
TX バッファ
924
TX コンテキスト MGR
TCB 1
TCB 2
TCB N
TCB N
926
フレーマ
GLM
リソース・コントローラ
934
NL ポート
936
938
フレーム DET
RX バッファ
RX DMA
RX データ
940
942
RX コンテキスト MGR
944
946
RCB 1
RCB 2
RCB N
RPY MSG 1
RPY MSG 2
RPY MSG N
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
A16
A17
A18
A19
A20
A21
A22a
A22b
A22c
A23
A24
A25
A26
A27
A28
A29
A30
A31
A32
A33
A34

FIG. 9

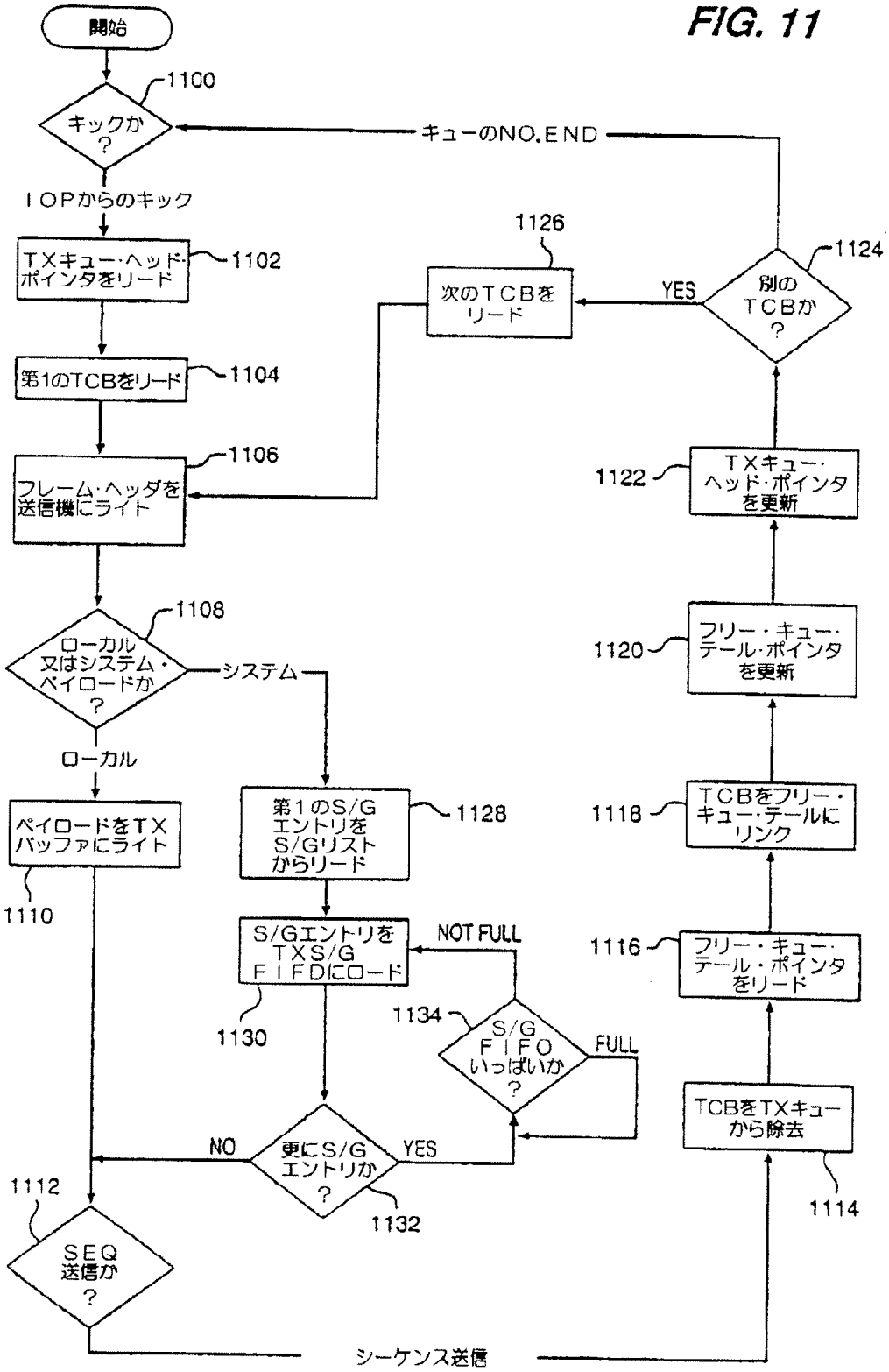
【図１０】

FREE_LISTの最上位
FREE_LIST
TAIL
次のフリー・フレームをここに追加
最後の入手可能なメッセージ・フレーム
n-1
...
3
2
次の入手可能なメッセージ・フレーム
HEAD
FREE_LISTのベース
POST_LISTの最上位
POST_LIST
次の満たされたフレームをここに追加
処理のための最後のメッセージ・フレーム
m-1
m-2
...
3
2
処理のための次のメッセージ・フレーム
HEAD
POST_LISTのベース

FIG. 10

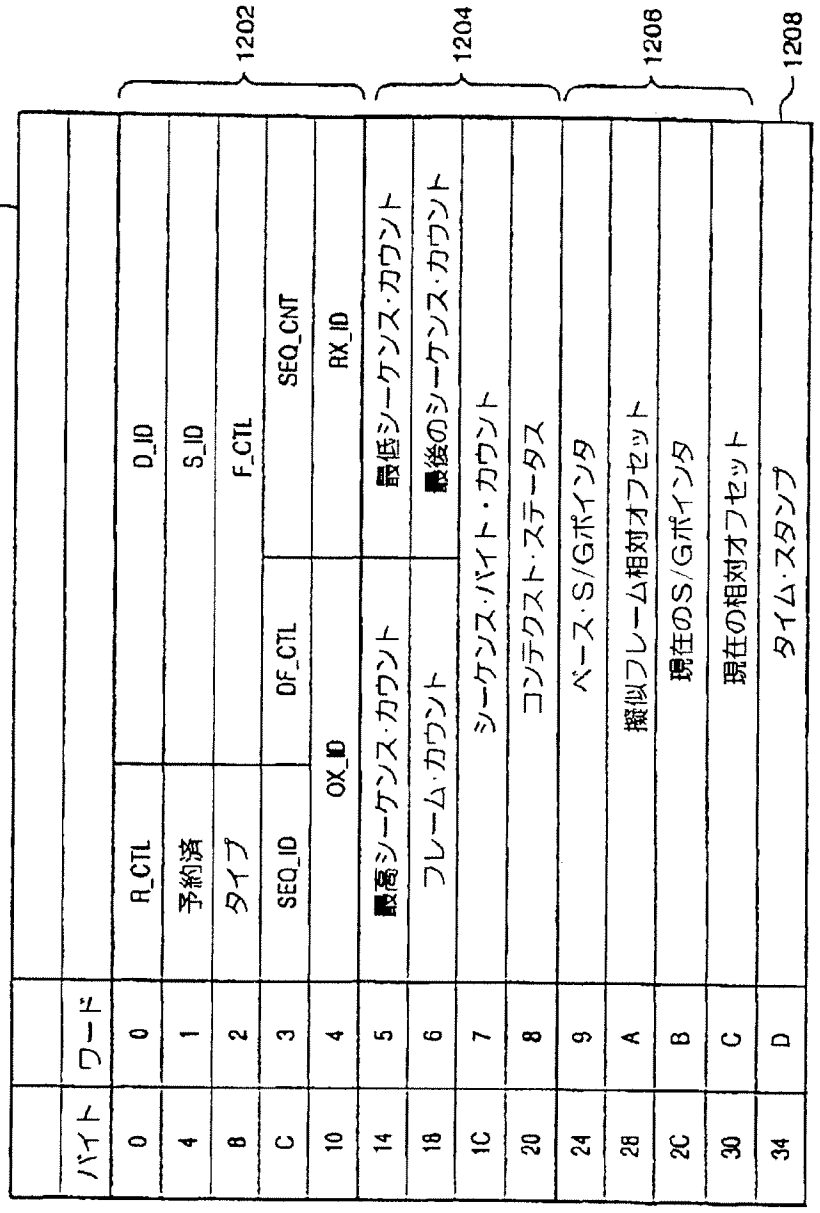
【図１１】

FIG. 11


開始
1100
キックか？
キューのNO.END
IOPからのキック
TXキュー・ヘッド・ポインタをリード
1102
第1のTCBをリード
1104
フレーム・ヘッダを送信機にライト
1106
1108
ローカル又はシステム・ペイロードか？
システム
ローカル
ペイロードをTXバッファにライト
1110
第1のS/Gエントリを S/Gリストからリード
1128
S/GエントリをTXS/G FIFOにロード
NOT FULL
1130
1134
S/G FIFO いっぱいか？
FULL
NO
更にS/Gエントリか？
YES
1132
1112
SEQ送信か？
シーケンス送信
1126
次のTCBをリード
YES
別のTCBか？
1124
1122
TXキュー・ヘッド・ポインタを更新
1120
フリー・キュー・テール・ポインタを更新
1118
TCBをフリー・キュー・テールにリンク
1116
フリー・キュー・テール・ポインタをリード
TCBをTXキューから除去
1114

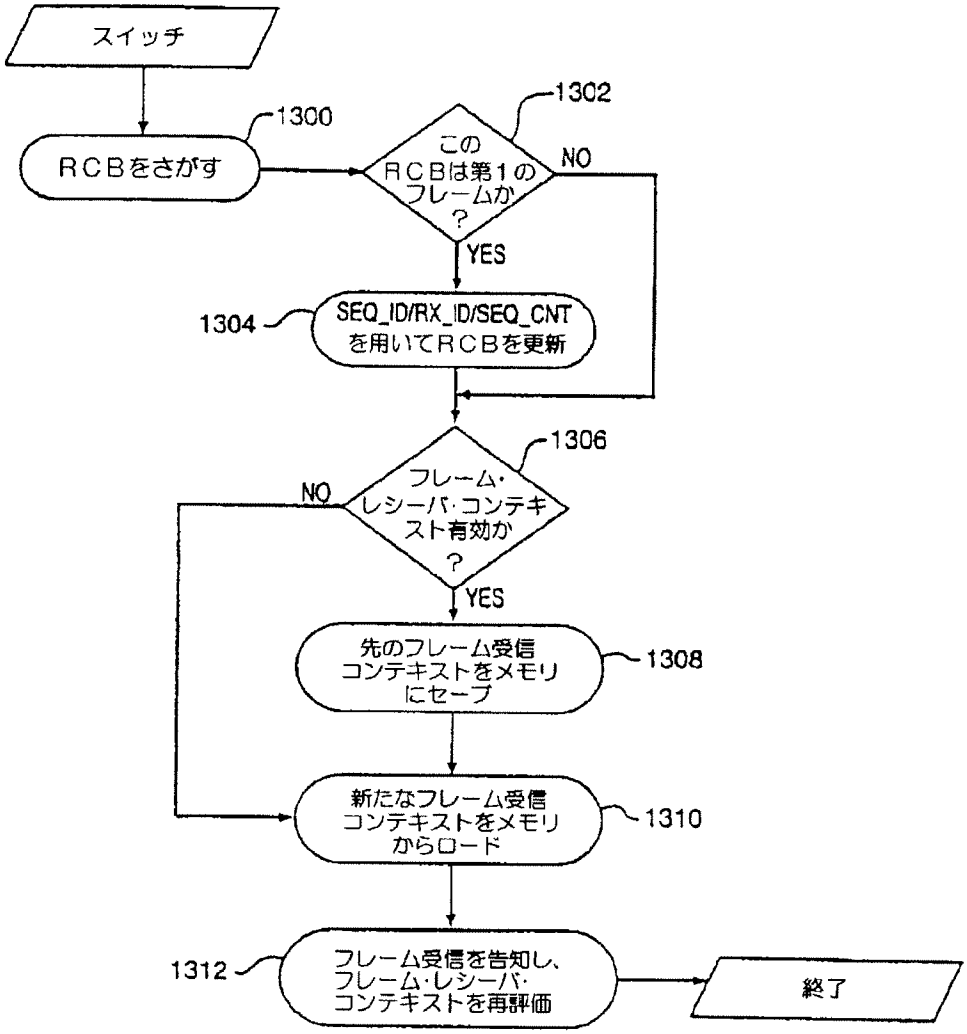
【図１２】

FIG. 12


1200
1202
1204
1206
1208
バイト
ワード
0
4
8
C
10
14
18
1C
20
24
28
2C
30
34
0
1
2
3
4
5
6
7
8
9
A
B
C
D
R_CTL
D_ID
予約済
S_ID
タイプ
F_CTL
SEQ_ID
DF_CTL
SEQ_CNT
OX_ID
RX_ID
最高シーケンス・カウント
最低シーケンス・カウント
フレーム・カウント
シーケンス・バイト・カウント
コンテクスト・ステータス
ベース・S/Gポインタ
擬似フレーム相対オフセット
現在のS/Gポインタ
現在の相対オフセット
タイム・スタンプ

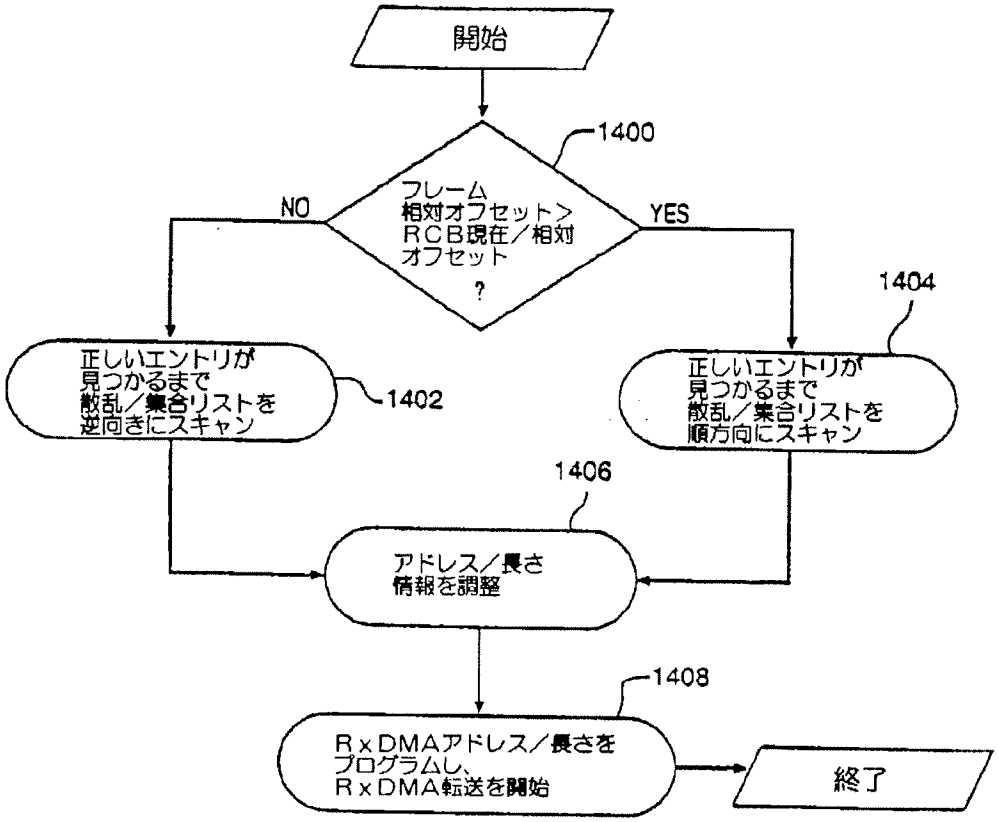
【図１３】

FIG. 13


スイッチ
1300
RCBをさがす
1302
このRCBは第1のフレームか？
NO
YES
1304
SEQ_ID/RX_ID/SEQ_CNTを用いてRCBを更新
1306
フレーム・レシーバ・コンテキスト有効か？
NO
YES
先のフレーム受信コンテキストをメモリにセーブ
1308
新たなフレーム受信コンテキストをメモリからロード
1310
1312
フレーム受信を告知し、フレーム・レシーバ・コンテキストを再評価
終了

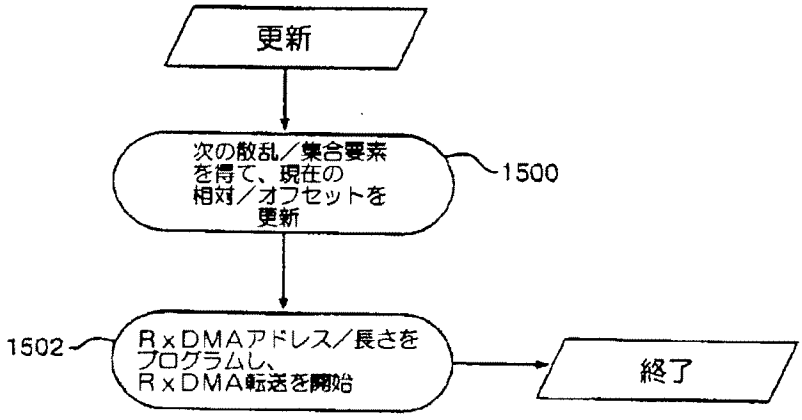
【図１４】

FIG. 14



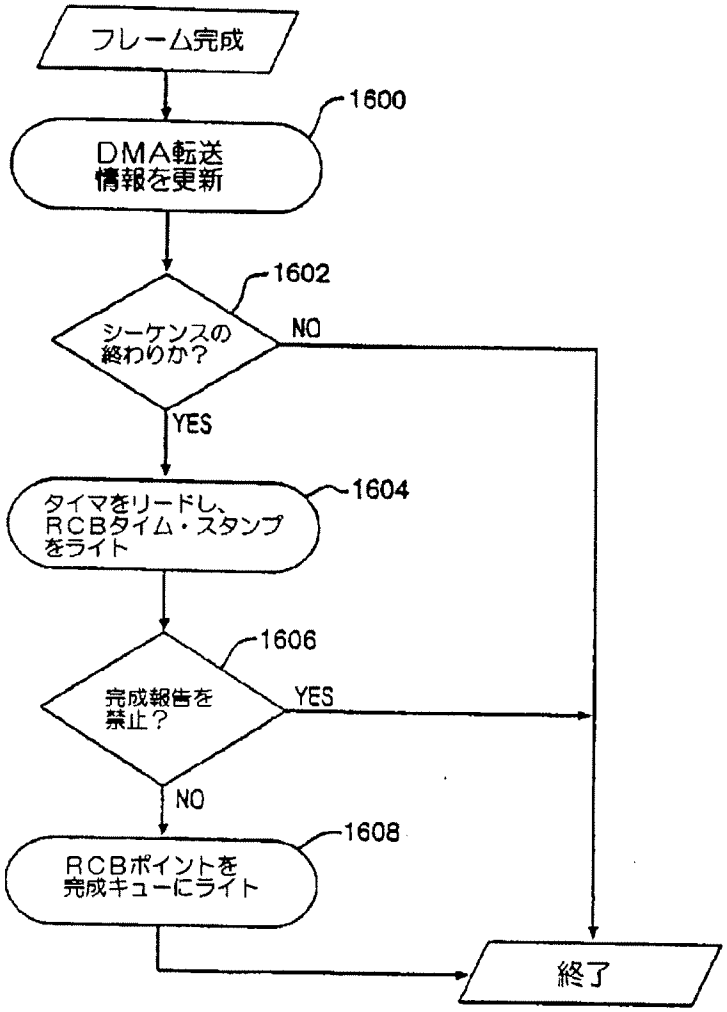
【図１５】

FIG. 15



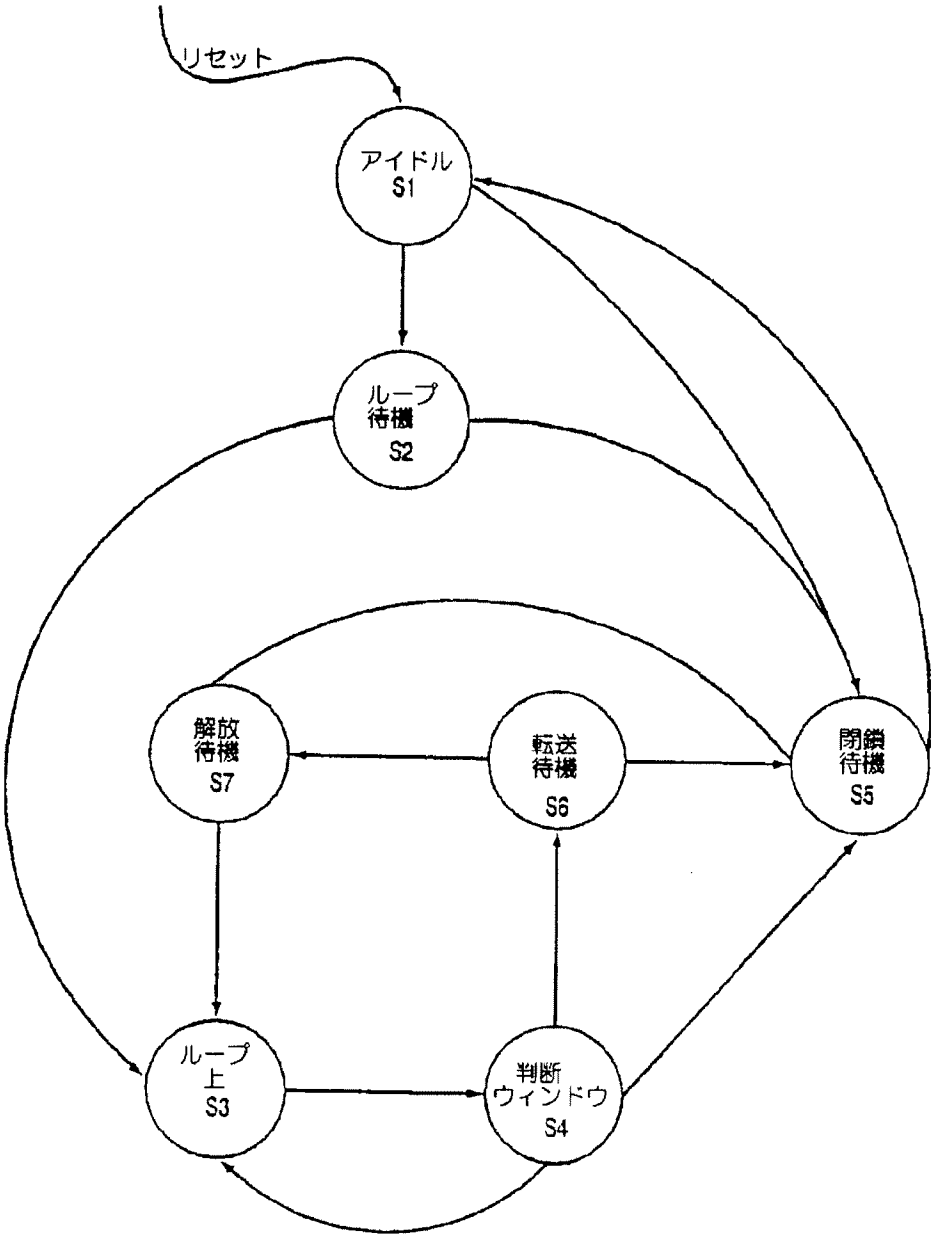
【図１６】

FIG. 16


フレーム完成
1600
DMA転送
情報を更新
1602
シーケンスの
終わりか？
NO
YES
1604
タイマをリードし、
RCBタイム・スタンプ
をライト
1606
完成報告を
禁止？
YES
NO
1608
RCBポイントを
完成キューにライト
終了

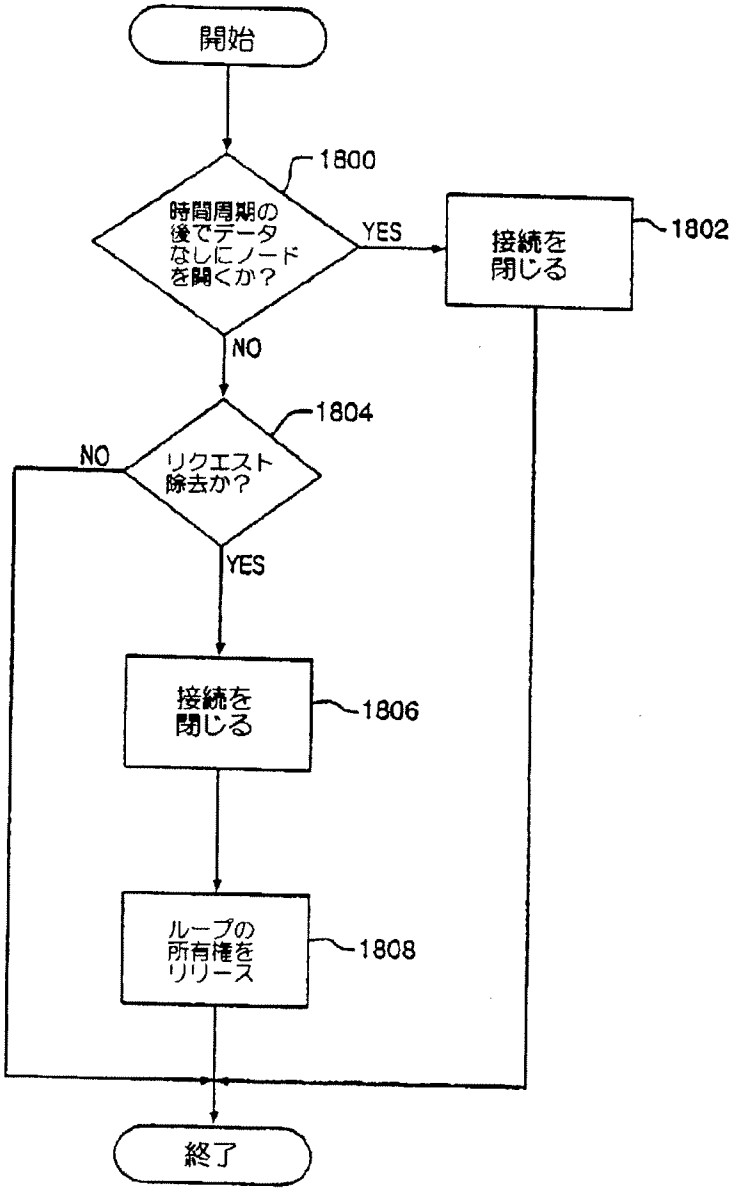
【図１７】

FIG. 17


リセット
アイドル
S1
ループ
待機
S2
解放
待機
S7
転送
待機
S6
閉鎖
待機
S5
ループ
上
S3
判断
ウィンドウ
S4

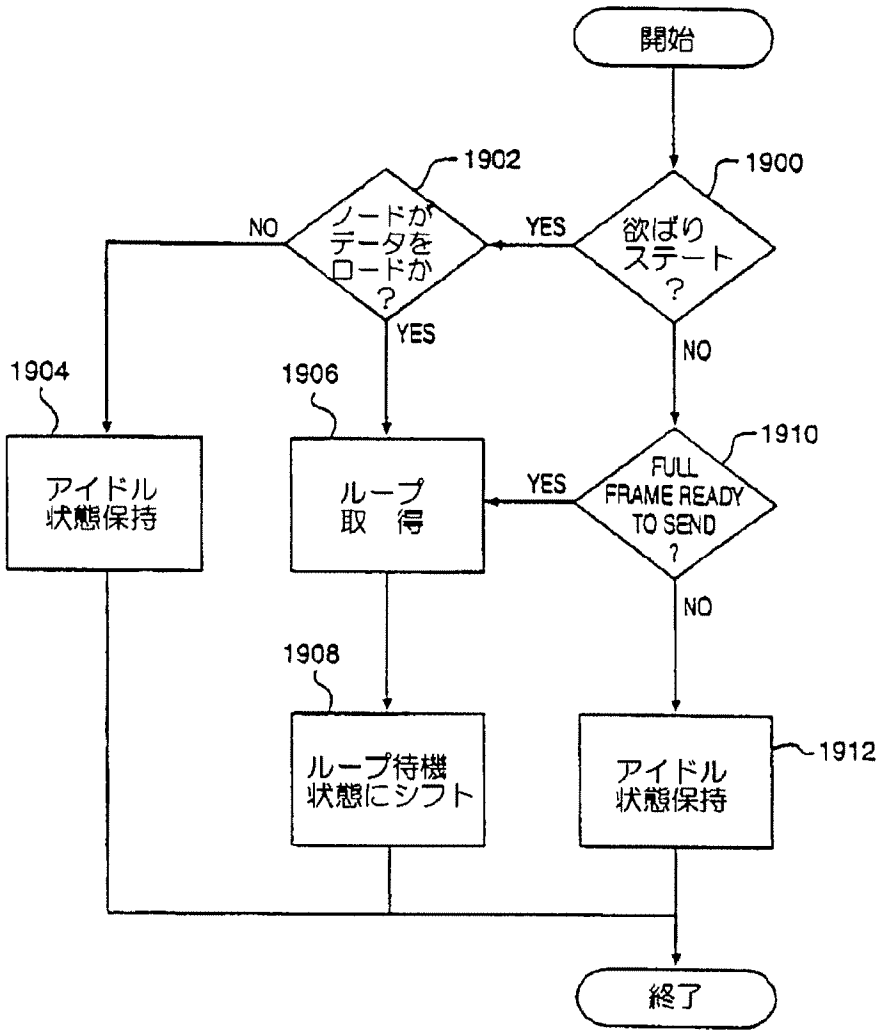
【図１８】

FIG. 18


開始
1800
時間周期の後でデータなしにノードを聞くか？
YES
接続を閉じる
1802
NO
1804
リクエスト除去か？
NO
YES
接続を閉じる
1806
ループの所有権をリリース
1808
終了

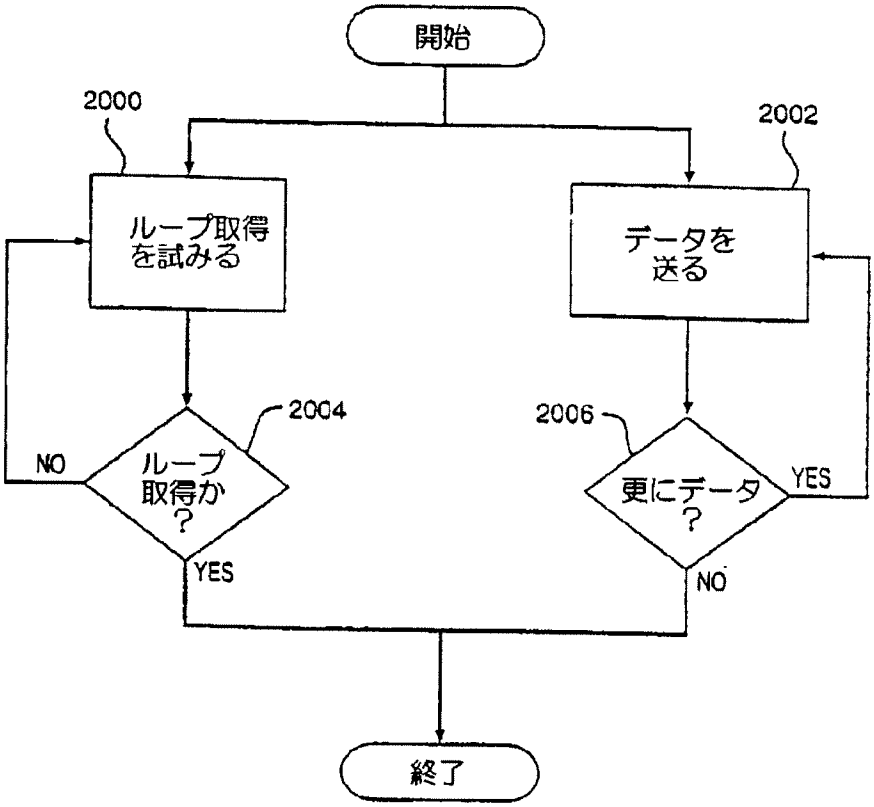
【図１９】

FIG. 19


開始
1900
欲ばり
ステート
?
1902
ノードが
データを
ロードか
?
NO
YES
YES
NO
1904
アイドル
状態保持
1906
ループ
取　得
1908
ループ待機
状態にシフト
1910
FULL
FRAME READY
TO SEND
?
YES
NO
1912
アイドル
状態保持
終了

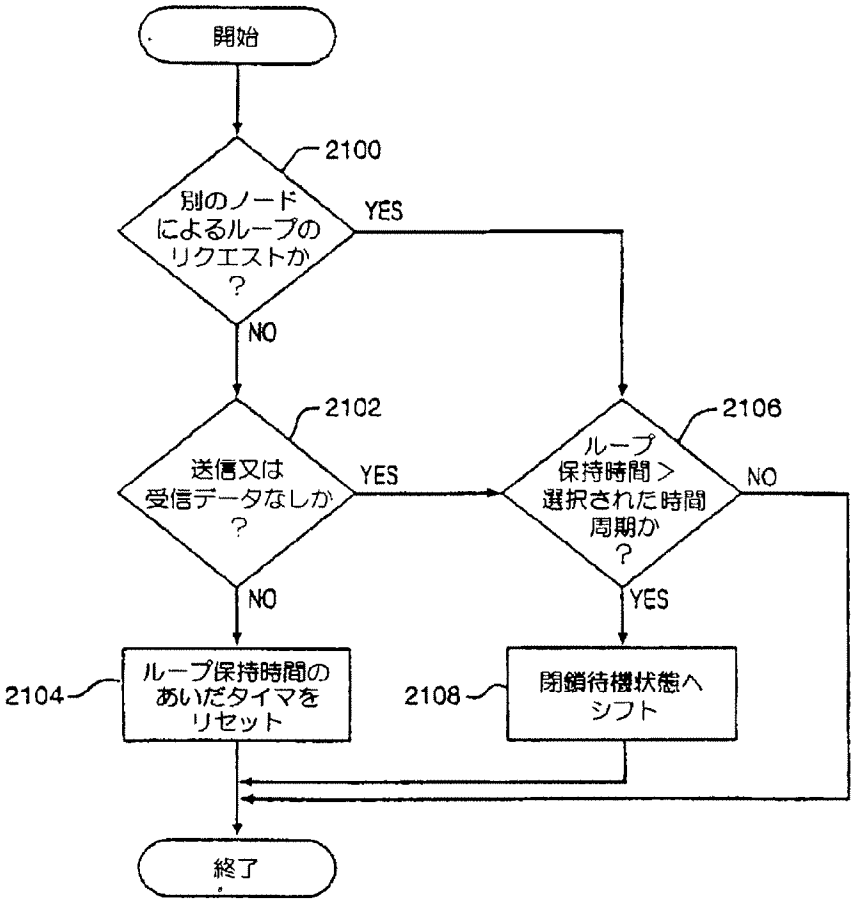
【図２０】

FIG. 20


開始
2000
ループ取得を試みる
2002
データを送る
2004
ループ取得か？
NO
YES
2006
更にデータ？
YES
NO
終了

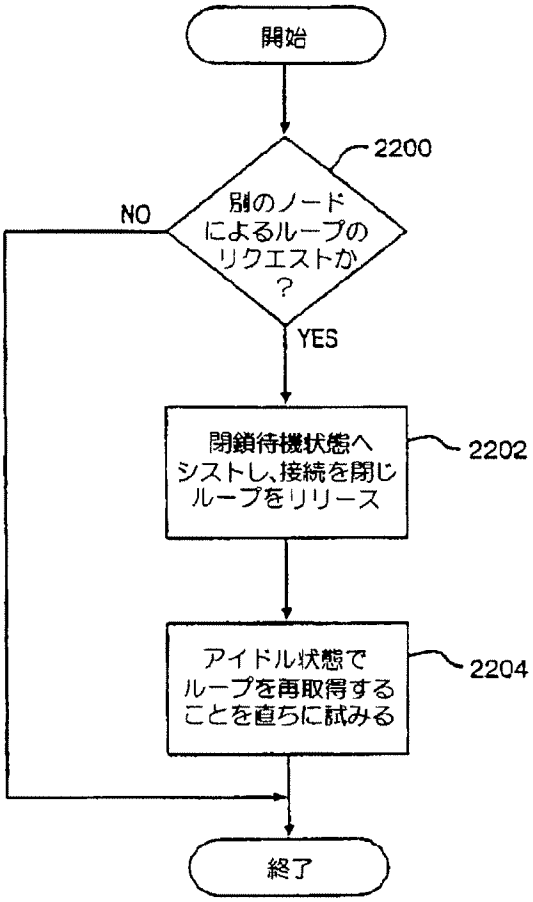
【図２１】

FIG. 21


開始
2100
別のノードによるループのリクエストか？
YES
NO
2102
送信又は受信データなしか？
YES
NO
2106
ループ保持時間＞選択された時間周期か？
NO
YES
2104
ループ保持時間のあいだタイマをリセット
2108
閉鎖待機状態へシフト
終了

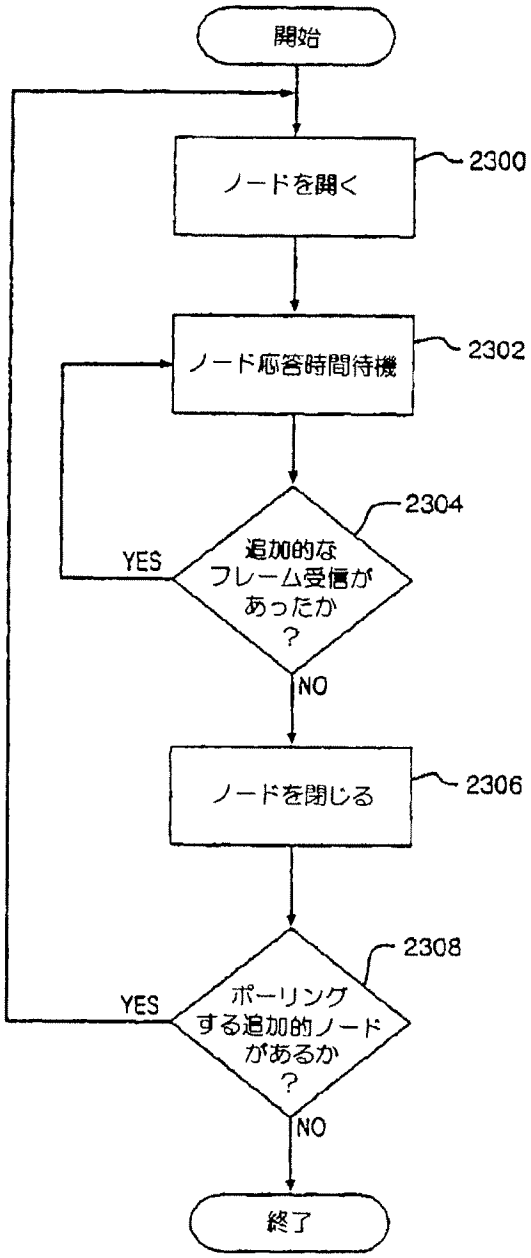
【図２２】

FIG. 22


開始
2200
別のノードによるループのリクエストか？
NO
YES
閉鎖待機状態へシストし、接続を閉じループをリリース
2202
アイドル状態でループを再取得することを直ちに試みる
2204
終了

【図２３】

FIG. 23


開始
ノードを開く
2300
ノード応答時間待機
2302
2304
追加的な
フレーム受信が
あったか
？
YES
NO
ノードを閉じる
2306
2308
ポーリング
する追加的ノード
があるか
？
YES
NO
終了

【手続補正書】特許法第１８４条の４第４項

【提出日】平成１１年１０月１４日（１９９９．１０．１４）

【補正内容】

［請求の範囲］の記載を次の通りに補正する。

『１．チップであって、

入力ポートと、

出力ポートと、

第１の処理ユニットであって、

このチップから離れて位置するホストからのリクエストを受信し、データを宛先に送信する第１の受信手段と、

前記データを受信する第２の受信手段と、

前記宛先と前記データとを、前記宛先への送信に先だって、このチップに接続されたメモリに記憶する記憶手段と、

を含む第１の処理ユニットと、

第２の処理ユニットであって、

宛先に送信するデータの存在を検出する検出手段と、

前記データと前記宛先とを用いて前記宛先への搬送のためのフォーマットに前記データをフォーマットするフォーマット手段と、

を含む第２の処理ユニットと、

を備えていることを特徴とするチップ。

２．請求項１記載のチップにおいて、前記第１の処理ユニットは、宛先に送られるデータのためのアクティビティのリストを作成し、前記検出手段は、前記リストを検査し前記宛先に送信するためのデータの存在を検出することを特徴とするチップ。

３．請求項１記載のチップにおいて、予測されるデータのリストが前記第１の処理ユニットによって発生され、前記第２の処理ユニットは、

データを遠隔ソースから受信する受信手段と、

前記データを記憶する識別手段と、

前記第１の処理ユニットに、いつすべてのデータが前記遠隔ソースから受信さ

れたかを指示する指示手段と、

を含むことを特徴とするチップ。

４．請求項１記載のチップ・アーキテクチャにおいて、前記第２の処理ユニットは埋め込み型プロセッサであることを特徴とするチップ・アーキテクチャ。

５．請求項１記載のチップにおいて、前記第２の処理ユニットはステート・マシンであることを特徴とするチップ。

６．請求項１記載のチップにおいて、前記第２の受信手段は、ホスト・メモリからこのチップに結合されたメモリにデータを転送する転送手段を備えていることを特徴とするチップ。

７．請求項１記載のチップにおいて、前記入力ポートは、ホスト・システム上のバスと通信するように構成され、他方で、前記出力ポートは、前記宛先と通信するように構成されていることを特徴とするチップ。

８．ホストから宛先にデータを転送するチップであって、

前記ホストから、前記宛先へのデータ転送のリクエストを受信し、メモリに前記宛先と前記データとのＩＤを記憶する第１の処理ユニットと、

前記メモリ内の前記データを検出し、前記第１の処理ユニットによる介入なしに前記ＩＤを用いて、前記データを前記宛先に転送する第２の処理ユニットと、

を備えていることを特徴とするチップ。

９．請求項８記載のチップにおいて、前記第２の処理ユニットは、前記データを前記宛先によって用いられるフォーマットに配置することを特徴とするチップ。

１０．請求項９記載のチップにおいて、前記ホストから受信されたデータは第１のフォーマットを有し、前記データは前記宛先によって用いられる第２のフォーマットにフォーマットされることを特徴とするチップ。

１１．請求項１０記載のチップにおいて、前記第１のフォーマットはＳＣＳＩプロトコルであることを特徴とするチップ。

１２．請求項１０記載のチップにおいて、前記第２のフォーマットはファイバ

・チャネル・プロトコルであることを特徴とするチップ。

１３．請求項８記載のチップにおいて、前記第２の処理ユニットは、前記宛先へのデータ転送を制御するコンテキスト・マネジャを含み、このチップ

は、更に、前記コンテキスト・マネジャによって制御されデータを前記宛先に転送する送信機を備えていることを特徴とするチップ。

１４．請求項１３記載のチップにおいて、

データを宛先から受信する受信機を更に備えており、

前記第２の処理ユニットは、前記データの受信を制御して、前記ホストへのデータの存在を前記第１の処理ユニットに指示することを特徴とするチップ。

１５．請求項８記載のチップにおいて、前記第２の処理ユニットは埋め込み型プロセッサであることを特徴とするチップ。

１６．請求項８記載のチップにおいて、前記第２の処理ユニットはステート・マシンであることを特徴とするチップ。

１７．請求項８記載のチップにおいて、前記第１の処理ユニットはプロトコル・エンジンであることを特徴とするチップ。

１８．請求項８記載のチップにおいて、前記第１の処理ユニットは、送信されるデータのリストを作成し、前記リストは、前記第２の処理ユニットによってデータを前記宛先に転送するのに用いられることを特徴とするチップ。

１９．請求項８記載のチップにおいて、

複数のフレームを受信する受信機を更に備えており、

前記第２の処理ユニットは、前記複数のフレームのそれぞれを、それらが前記受信機によって受信される際に処理し、前記複数のフレームのすべてがいつ受信されたかを判断して、前記複数のフレームのすべてが受信されたという指示を提供することを特徴とするチップ。

２０．チップであって、

データを宛先に転送するというホストからのリクエストを受信するプロトコル・エンジンと、

前記プロトコル・エンジンに接続されており、前記ホストからの前記データを

受信するデータ・ムーバと、

前記プロトコル・エンジンと前記データ・ムーバとに接続されており、前記データ・ムーバによって受信されたデータを前記プロトコル・エンジンに

よる介入なしに前記宛先に転送するトランスポート制御ユニットと、

を備えていることを特徴とするチップ。

２１．請求項２０記載のチップにおいて、前記データ・ムーバはバスによって前記ホストに接続されていることを特徴とするチップ。

２２．請求項２０記載のチップにおいて、前記トランスポート制御ユニットは、

前記データ・ムーバに接続されており、前記ホストから受信されたデータを前記宛先に送信する送信機と、

前記データ・ムーバに接続されており、ソースから前記ホストへのデータを受信する受信機と、

前記受信機と前記送信機とに接続されており、前記送信機によるデータの送信と前記受信機によるデータの受信とを制御する処理ユニットと、

を含むことを特徴とするチップ。

２３．請求項２２記載のチップにおいて、前記処理ユニットは埋め込み型プロセッサであることを特徴とするチップ。

２４．ホストから宛先にデータを転送する装置であって、

第１のプロセッサであって、

前記ホストから前記宛先にデータを転送するリクエストを受信する第１の受信手段であって、前記リクエストは、それぞれが、前記宛先のＩＤを含む、第１の受信手段と、

前記宛先に転送するデータを受信する第２の受信手段と、

前記宛先と第１のプロトコルを有する前記データとをメモリに記憶する記憶手段と、

を含む第１のプロセッサと、

第２のプロセッサであって、

第１のフォーマットの前記データを第２のフォーマットにフォーマットしフォーマットされたデータを形成するフォーマット手段と、

前記フォーマットされたデータを前記記憶手段に記憶された前記宛先を用いて転送する転送手段と、

を含む第２のプロセッサと、

を備えていることを特徴とする装置。

２５．請求項２４記載の装置において、前記第１のプロトコルはＳＣＳＩプロトコルであることを特徴とする装置。

２６．請求項２４記載の装置において、前記第２のプロトコルはファイバ・チャネル・プロトコルであることを特徴とする装置。

２７．請求項２４記載の装置において、前記リクエストは、ホスト・プロセッサがデータ転送のためのリクエストを配置するリクエスト・キューを用いてなされることを特徴とする装置。

２８．請求項２４記載の装置において、前記第１のプロセッサは、前記宛先に送られるデータのリストを作成するリスト化手段を更に含み、前記第２のプロセッサにおける前記転送手段は、前記リストを用いてデータを前記宛先に転送することを特徴とする装置。

２９．請求項２４記載の装置において、遠隔ソースからデータを受信するポートを更に含んでおり、前記第２のプロセッサは、

前記枝なっくソースから受信された前記第２のプロトコルを有するデータを記憶する記憶手段と、

前記第２のプロトコルを有する前記データを前記第１のプロトコルにフォーマットする第２のフォーマット手段と、

前記第１のプロセッサにデータが前記遠隔ソースから受信されたことを指示する指示手段と、

を更に含むことを特徴とする装置。

３０．チップであって、

ホスト上のバスからのデータを送信及び受信するバス・インターフェース・ユ

ニットと、

前記バス・インターフェース・ユニットに接続されており、前記バスに結合されたホスト・メモリからこのチップに結合されたローカル・メモリへの情報の転送を管理し、データをデバイスに送信するのに用いられた送信アクティビティのリストを発生するプロトコル・エンジンと、

前記デバイスへの通信リンクのためのインターフェースを提供するリンク・コントローラと、

前記リンク・コントローラに接続されており、前記リンク・コントローラによる前記デバイスへの転送のためのフォーマットへのデータのフォーマットを管理する送信機と、

前記リンク・コントローラに接続されており、前記リンク・コントローラによって受信されるデータを管理する受信機と、

前記プロトコル・エンジンに結合されており、送信アクティビティの前記リストを検討し、前記送信機を用いて前記リストに基づきデータの転送を実行して、前記受信機によって受信されるデータを処理するコンテキスト・マネジャと、

を備えていることを特徴とするチップ。

３１．請求項３０記載のチップにおいて、前記プロトコル・エンジンは埋め込み型プロセッサであることを特徴とするチップ。

３２．請求項３１記載のチップにおいて、前記コンテキスト・マネジャは埋め込み型プロセッサであることを特徴とするチップ。

３３．請求項３１記載のチップにおいて、前記コンテキスト・マネジャはステート・マシンであることを特徴とするチップ。

３４．請求項３０記載のチップにおいて、前記データはフレームの形式であり、前記コンテキスト・マネジャは異なるソースから順不同で受信されるフレームを処理することを特徴とするチップ。

３５．請求項３０記載のチップにおいて、前記コンテキスト・マネジャは、送信アクティビティの前記リストを検討し、送信アクティビティの前記リストから第１の項目を読み出すことによってデータ転送を実行し、前記項目に対するデー

タがローカル・データであるかどうかを判断し、前記データがローカルであるという判断に応答して、前記データを前記ローカル・メモリから前記送信機に書き込むことを特徴とするチップ。

３６．請求項３５記載のチップにおいて、前記コンテキスト・マネジャは、前記項目に対するデータがシステム・データであるという判断に応答して、

バスを介して、前記ホスト・メモリからのデータを送信機にロードすることを特徴とするチップ。

３７．請求項３６記載のチップにおいて、前記送信機は、データとコンテキスト情報とを記憶する記憶装置と、前記記憶装置に接続されたフレーマとを含むことを特徴とするチップ。

３８．請求項３５記載のチップにおいて、送信アクティビティの前記リストは、複数の送信制御ブロックであることを特徴とするチップ。

３９．チップであって、

第１のバスからホストへデータを送信及び受信するインターフェースを有するバス・インターフェース・ユニットと、

前記バス・インターフェース・ユニットに接続されており、ホスト・メモリからこのチップに結合されたメモリへの情報の転送を管理し、データ転送のリクエストに応答し、複数の動作モードを有するプロトコル・エンジンであって、

前記プロトコル・エンジンがホスト・メモリの中にあるデータ・ブロックを転送するリクエストを検出する第１の動作モードと、

前記リクエストの検出に応答する第２の動作モードであって、前記プロトコル・エンジンは、前記データ・ブロックを前記ホスト・メモリからローカル・メモリの中に移動させ、前記送信ブロックは、前記データ・ブロックと前記データ・ブロックを送信するのに用いられる情報とを含む、第２の動作モードと、

を備えたプロトコル・エンジンと

前記送信ブロックを検出し、前記送信ブロックを送信機に送信する転送エンジンであって、前記送信機は、前記送信ブロックを受信してデータ・ブロックを送信のためのフォーマットに配置し、前記フォーマットは前記情報から識別される

、転送エンジンと、

前記送信機に接続されており、前記デバイスへの通信リンクのためのインターフェースを提供するリンク・コントローラと、

を備えていることを特徴とするチップ。

４０．請求項３９記載のチップにおいて、前記コンテキスト情報は、宛先への経路のＩＤを含むことを特徴とするチップ。

４１．請求項３９記載のチップにおいて、前記リンク・コントローラは、ファイバ・チャネル仲裁ループによる通信のために構成されていることを特徴とするチップ。』

【国際調査報告】

INTERNATIONAL SEARCH REPORT

Inter nal Application No
PCT/US 99/06772

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 H04L29/06 G06F13/12

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 H04L G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	QLOGIC CORPORATION: "ISP2100 Intelligent Fibre Channel Processor" DATA SHEET, 'Online! 29 July 1997 (1997-07-29), XP002111444 Retrieved from the Internet: <URL:http://qlogic.qlc.com/products/pdf/data_sheets/83210-580-00c.pdf> 'retrieved on 1999-08-04! the whole document ---	1-5
X	SMITH J A ET AL: "TACHYON: A GIGABIT FIBRE CHANNEL PROTOCOL CHIP" HEWLETT-PACKARD JOURNAL, vol. 47, no. 5, 1 October 1996 (1996-10-01), pages 99-112, XP000631672 page 101, left-hand column, line 6 - page 111, left-hand column, line 24 --- -/--	1-3,5

[X] Further documents are listed in the continuation of box C

[ ] Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
5 August 1999	17/08/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Eraso Helguera, J

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Inter nal Application No
PCT/US 99/06772

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	CHILD J: "I2O GEARS UP FOR EMBEDDED USE" COMPUTER DESIGN, vol. 36, no. 8, 1 August 1997 (1997-08-01), page 15/16, 18 XP000735779 ISSN: 0010-4566 page 18, right-hand column, paragraph 2	4

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

フロントページの続き

(72)発明者 ジョンソン，スティーヴン・エム
アメリカ合衆国コロラド州80906，コロラド・スプリングズ，アラパホ・ドライブ 522
(72)発明者 アダムズ，ジョン・エム
アメリカ合衆国コロラド州80919，コロラド・スプリングズ，ライフル・サークル 6430
(72)発明者 リーバー，マーク・エイ
アメリカ合衆国ジョージア州30005，アルファレッタ，プリザーブ・レイン 12495

【要約の続き】
断する。更に、リンク制御ユニットが提供され、そこでは、ホストがループに接続されるときに、ループ管理が提供される。ループの管理には、ホストとループ上の他のノードとによってデータが受信及び送信される状態に応答して、ループの取得を開始しループの解放を開始するメカニズムの実現が含まれる。

FIG. 8B